UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS, ELECTRÓNICA E INDUSTRIAL

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y COMUNICACIONES

Tema:

SISTEMA DE MANIPULACIÓN Y MONITOREO DEL ROBOT SCORBOT MEDIANTE SEÑALES ELECTROENCEFALOGRÁFICAS (EEG).

Trabajo de Titulación Modalidad: Proyecto de Investigación, presentado previo la obtención del título de Ingeniero en Electrónica y Comunicaciones.

ÁREA: Electrónica

LÍNEA DE INVESTIGACIÓN: Tecnologías de Información y Sistemas de Control

AUTOR: Walter David Cunalata Velasco

TUTOR: Ing. Franklin Salazar Logroño, Mg.

AMBATO – ECUADOR

agosto-2021

APROBACIÓN DEL TUTOR

En calidad de tutor del Trabajo de Titulación con el tema: SISTEMA DE MANIPULACIÓN Y MONITOREO DEL ROBOT SCORBOT MEDIANTE SEÑALES ELECTROENCEFALOGRÁFICAS (EEG), desarrollado bajo la modalidad Proyecto de Investigación por el señor Walter David Cunalata Velasco, estudiante de la Carrera de Ingeniería en Electrónica y Comunicaciones, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, de la Universidad Técnica de Ambato, me permito indicar que el estudiante ha sido tutorado durante todo el desarrollo del trabajo hasta su conclusión, de acuerdo a lo dispuesto en el Artículo 15 del Reglamento para obtener el Título de Tercer Nivel, de Grado de la Universidad Técnica de Ambato, y el numeral 7.4 del respectivo instructivo

Ambato, agosto de 2021

Firmado electrónicamente por:
FRANKLIN WILFRIDO SALAZAR LOGRONO

Ing. Mg. Franklin Salazar

TUTOR

AUTORÍA

El presente Proyecto de Investigación titulado: SISTEMA DE MANIPULACIÓN Y MONITOREO DEL ROBOT SCORBOT MEDIANTE SEÑALES ELECTROENCEFALOGRÁFICAS (EEG), es absolutamente original, auténtico y personal. En tal virtud, el contenido, efectos legales y académicos que se desprenden del mismo son de exclusiva responsabilidad del autor.

Ambato, agosto de 2021

Firmado electrónicamente por:
WALTER DAVID CUNALATA VELASCO

_____

Walter David Cunalata Velasco

1804256509

AUTOR

APROBACIÓN DEL TRIBUNAL DE GRADO

En calidad de par calificador del Informe Final del Trabajo de Titulación presentado por el señor Walter David Cunalata Velasco, estudiante de la Carrera de Ingeniería en Electrónica y Comunicaciones, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, bajo la Modalidad Proyecto de Investigación, titulado SISTEMA DE MANIPULACIÓN Y MONITOREO DEL ROBOT SCORBOT MEDIANTE SEÑALES ELECTROENCEFALOGRÁFICAS (EEG), nos permitimos informar que el trabajo ha sido revisado y calificado de acuerdo al Artículo 17 del Reglamento para obtener el Título de Tercer Nivel, de Grado de la Universidad Técnica de Ambato, y al numeral 7.6 del respectivo instructivo. Para cuya constancia suscribimos, conjuntamente con la señora Presidenta del Tribunal.

Ambato, agosto 2021.

Firmado electrónicamente por:
ELSA PILAR URRUTIA

Ing. Elsa Pilar Urrutia Mg.
PRESIDENTA DEL TRIBUNAL

Firmado electrónicamente por:
EDGAR PATRICIO CORDOVA CORDOVA

Ing. Patricio Córdova Mg.
DOCENTE CALIFICADOR

Firmado electrónicamente por:
ANDREA PATRICIA SANCHEZ ZUMBA

Ing. Andrea Sánchez Mg.
DOCENTE CALIFICADOR

DERECHOS DE AUTOR

Autorizo a la Universidad Técnica de Ambato, para que haga uso de este Trabajo de Titulación como un documento disponible para la lectura, consulta y procesos de investigación.

Cedo los derechos de mi Trabajo de Titulación en favor de la Universidad Técnica de Ambato, con fines de difusión pública. Además, autorizo su reproducción total o parcial dentro de las regulaciones de la institución.

Ambato, agosto de 2021

Walter David Cunalata Velasco

1804256509

AUTOR

DEDICATORIA

A Dios, por haberme concedido el privilegio de existir y el don de conocerle.

A mis padres quienes sembraron las bases morales y los conceptos de amor necesarios para construir una vida.

A mis maestros, que ejercitando mis facultades, me abrieron un nuevo horizonte, un nuevo camino, con meta y destino promisorio. Que con honestidad y sapiencia moldearon mi ser y me concedieron el tesoro más codiciado de la educación.

A Gabriela Valverde, la persona especial que me apoyo todo el tiempo y le ha dado a mi vida un valor diferente.

Walter David Cunalata Velasco

AGRADECIMIENTO

Un agradecimiento sincero al Ingeniero Franklin Salazar, cuya ayuda y conocimiento fueron fundamentales para realización de este trabajo.

A mis docentes calificadores Ing. Patricio Córdova y Ing. Andrea Sánchez quienes revisaron el manuscrito de este trabajo y contribuyo a mejorar notablemente su estructura y su contenido.

A mis amigos, Paulo y Patricio que de una u otra forma me escucharon o dieron unas palabras que cambiaron el rumbo de mi carrera universitaria.

A todos quienes hacen la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, que sembraron en mi vida bases profundas de dedicación y sacrificio.

Walter David Cunalata Velasco

ÍNDICE GENERAL DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

Una de las principales herramientas que contribuye a una mejor calidad vida, a pesar de las dificultades motoras es la tecnología, esta promueve la creación e investigación de nuevos equipos, el costo de manipular robots industriales a través de software propietario es muy alto y existen limitaciones por la imposibilidad de integrar nuevas tecnologías de comunicación.

El objetivo de esta investigación fue desarrollar un sistema de control y monitoreo del Robot Scorbot mediante las señales electroencefalográficas (EEG), que tiene la capacidad de manipular las articulaciones del Robot Scorbot, utilizando las señales eléctricas que proporciona la actividad cerebral, para esto se utilizó el casco Neurosky y el emotiv insight, cual recepta los impulsos de los electrodos, convirtiendo en señales que son procesadas por el software matemático Matlab. En la etapa de control y monitoreo se desarrolló una interfaz gráfica, en donde el usuario pueda entrenar y acondicionar el sistema dependiendo de sus discapacidades físicas, mentales e intelectuales.

La etapa de control electrónico se realizó en una raspberry Pi3 B+, utilizando el protocolo MQTT para la recepción y envió de información entre las etapas del sistema tanto como para la comunicación con el Robot Scorbot, silla de ruedas o el brazo robótico Eezy, por otra parte se utilizó ordenadores de bajo costo para el control de los diferentes movimientos que tendrá cada uno de los dispositivos, como también para un mejor monitoreo se ensambló un prototipo en 3D con la ayuda del software Unity, para visualización de los movimientos del Robot Scorbot.

Palabras clave: neurosky, raspberry pi, mqtt, electroencefalograficas, scorbot, emotiv insight

ABSTRACT

One of the main tools that contributes to a better quality of life, despite motor difficulties is technology, this promotes the creation and research of new equipment, the cost of manipulating industrial robots through proprietary software is very high and there are limitations due to the impossibility of integrating new communication technologies.

The objective of this research was to develop a control and monitoring system of the Scorbot Robot using electroencephalographic signals (EEG), which has the ability to manipulate the joints of the Scorbot Robot, using the electrical signals provided by brain activity. the Neurosky helmet and the emotiv insight, which receives the impulses from the electrodes, converting them into signals that are processed by Matlab mathematical software. In the control and monitoring stage, a graphical interface was developed, where the user can train and condition the system depending on their physical, mental and intellectual disabilities.

The electronic control stage was carried out on a raspberry Pi3 B +, using the MQTT protocol to receive and send information between the stages of the system as well as for communication with the Scorbot Robot, wheelchair or the Eezy robotic arm, on the other. In part, low-cost computers were used to control the different movements that each of the devices will have, as well as for better monitoring, a 3D prototype was assembled with the help of the Unity software, to visualize the movements of the Scorbot Robot.

Keywords: neurosky, raspberry pi, mqtt, emotiv insight, scorbot electroencephalography.

CAPÍTULO I
MARCO TEÓRICO

1.1 Tema

SISTEMA DE MANIPULACIÓN Y MONITOREO DEL ROBOT SCORBOT MEDIANTE SEÑALES ELECTROENCEFALOGRÁFICAS (EEG).

1.2 Antecedentes Investigativos

Los presentes antecedentes investigativos están diseñados para crear un análisis de investigaciones previas que permitan determinar un enfoque metodológico, especificando la relevancia y diferencias con el trabajo propuesto y las circunstancias que lo justifican; para ello se ha tomado datos de diferentes revistas, paper, tesis y trabajos científicos.

En el año 2020, la Revista IEEE publicó un artículo proveniente de Sharjah, United Arab Emirates denominado "Silla de ruedas EEG para personas con parálisis parcial o total", propuesto por Mariam AlAbboudi, Maitha Majed, Fatima Hassan, Ali Bou Nassif; presenta un sistema basado en la técnica llamada BCI de interfaz cerebro-computadora, que proporciona una conexión entre el cerebro y el controlador traduciendo los comandos cerebrales individuales en comportamientos. Esto se basa en el uso de los auriculares Emotiv Epoc que adquieren las señales del cerebro, caracterizado en el uso de electrodos colocados en el cuero cabelludo con una pasta conductora que pretende ayudar a personas que padecen parálisis parciales o totales mediante la manipulación de una silla de ruedas controlada por las señales electroencefalográficas [1].

Ivan N. Zamora, Diego S. Benitez, and Manuel S. Navarro, en el año 2019, desarrollaron su investigación "En el control de un brazo robótico utilizan señales sin procesar adquiridas del EMOTIV Insight NeuroHeadset", Este documento propone el uso del servicio de interfaz de programación (API) de cortex para obtener señales sin procesar del Emotiv Insight NeuroHeadset, utilizando un software de adquisición de

datos basado en Python fue desarrollado para usar el servicio a través de la transmisión de datos en un Websocket. Con en ello se puede orientar a la interacción completa con el brazo robótico controlado por una raspberry PI 3 usando el protocolo MQTT [2].

En el año 2019, la Revista IEEE publicó un artículo proveniente de DaLian, China y denominado "Sistema de control de una casa inteligente controlado por ondas cerebrales", propuesto por Li Yingda, Zhang Fuyan, Yang Yiqing; desarrollando un diseño de un sistema de control inteligente para una casa mediante la actividad eléctrica generada por el cerebro y haciendo uso del sensor TGAM producido por NeuroSky Shenneian technology, que detecta las señales de ondas cerebrales del lóbulo frontal. Obteniendo como resultado el control de la casa, ajustando a las necesidades particulares de las personas que viven con enfermedades crónicas a demás facilitar el día a día la rehabilitación de pacientes con discapacidad de extremidades [3].

En el año 2017 en el repositorio de la Universidad Técnica de Ambato de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial en la Carrera de Ingeniería en Electrónica y Comunicaciones, se encontró un trabajo con el tema "Prototipo de prótesis de un brazo con 12 GDL controlada mediante ondas cerebrales", en donde se presenta el diseño y construcción de una prótesis de un brazo que cuenta con 12 grados de libertad y es controlado mediante la actividad eléctrica generada por el cerebro dichas señales son adquiridas por electrodos EEG; además, un circuito de adquisición de señales electroencefalográficas (EEG). Este proyecto se realizó con el fin de contar con un prototipo de prótesis que emule los movimientos del miembro superior ausente con un grado de movilidad aceptable lo cual ayudará a las personas que hayan perdido una extremidad[4].

En el año 2017, la revista IEEE, se publicó una investigación proveniente de Bucharest Romania, denominada "Control de un robot humanoide con las señales electroencefalogramas (EEG)", presento un sistema basado en la técnica llamada BCI de interfaz cerebro-computadora, que propone un control de un robot humanoide a través de las señales electroencefalografícas (EEG) empleando el software y los auriculares Emotiv Epoc para la recepción y procesamiento de las señales,

consiguiendo así un control mediante la lectura de las ondas cerebrales para realizar tareas uniformes, desagradables y /o peligrosas. [5].

En el año 2016, Sim Kok Swee, Lim Zheng You , en la ciudad Melaka, Malaysia, en su investigación con el tema "Análisis rápido de Fourier (FFA) para las ondas cerebrales que manipulan una silla de ruedas ", realizaron un análisis rápido de fourier (FFA) con las ondas cerebrales basadas en la clasificación del electroencefalogramas (EEG), el método empleado es el Brain-Computer Interfaz (BCI) permitiendo la comunicación entre el cerebro y la silla de ruedas eléctrica, utilizando el auricular Emotiv EPOC + que tiene 14 canales y es capaz de transmitir las señal de EEG de forma inalámbrica a la PC, para el control de la silla utilizaron una placa microcontroladora Arduino Uno, obteniendo como resultado la silla de ruedas controlada por las señales eléctricas [6].

En el año 2014, Akinori Sakaguchi y Takashi Takimoto , en la Ciudad de Gyeonggido en Korea, realizaron la Investigación de "Desarrollo de un sistema de soporte para las operaciones de helicópteros multirrotor utilizando señales electroencefalografícas" el proyecto se desarrolló mediante tres etapas: en la primera etapa permite la recolección de datos emitidas por las señales electroencefalograficas (EEG) captada por el sensor EPOC, mientras que en la segunda etapa se encargado de la manipulación del helicóptero multirrotor mediante los módulos MPU-6000, HMC5883, MS561I y en la tercera etapa realiza la comunicación inalámbrica para la transmisión y recepción de información a través del módulo ZigBee. Teniendo como resultados diferentes rangos de frecuencia que varían dinámicamente de acuerdo a factores internos de ser humano, es decir que el nivel de intensidad de las ondas aumenta cuando se enfoca en un solo pensamiento operando de manera normal el helicóptero multirrotor y disminuye cuando se distrae detectando condiciones anormales en la operación [11].

1.2.1 Contextualización del problema

El continuo desarrollo de la robótica tiene un gran impacto a escala global, haciendo que la humanidad avance hacia la investigación permanente, generando así beneficios directos en diferentes campos como la medicina, la industria, y la educación. El campo

industrial utiliza manipuladores robóticos y sus aplicaciones para mejorar la eficiencia del proceso a través de interfaces hombre-máquina. [8].

En los últimos años, el desarrollo de aplicaciones industriales es importante, y con él surge la necesidad de incorporar maquinas programadas para realizar actividades humanas de alto riesgo y de difícil acceso, de ahí la idea de sustituirlas por manipuladores industriales. Por ello, es necesario monitorear los procesos industriales, donde los robots son participantes, y recolectando información estadística, datos relacionados con procesos industriales, motores y máquinas [9].

Según estadísticas de la Federación Internacional de Robótica, en América Latina se han instalado un total de 42.041 robot industriales. Los robots también se utilizan en la producción de plásticos, productos químicos, metales y alimentos, aunque la popularidad de la robótica es mucho menor en estas áreas, tiene ventajas importantes al realizar tareas con mayor intensidad, precisión y velocidad, remplazando a las actividades peligrosas que ponen en riesgo la integridad de las personas. Pese a su necesidad los costos de implementación son demasiados elevados haciendo imposible que la mayoría de las industrias puedan acceder a ella [10].

En Ecuador, uno de los principales objetivos es desarrollar la robótica en determinados campos, tales como: agricultura, agroindustria, biotecnología, procesamiento de metales, medicina, etc., por esto es necesario contar con profesionales e investigaciones aplicadas que puedan desarrollarse en las instituciones de educación superior. Para la industria, las adquisiciones son difíciles debido a los altos costos, por lo que pocas empresas pueden implementar con éxito manipuladores y robots [11].

La Ingeniería en Electrónica y Comunicaciones al ser la encargada de formar profesionales con la capacidad de crear investigaciones aplicadas y aportar con ideas innovadoras para el desarrollo del país, en base a las problemáticas constatadas surge la necesidad de crear un sistema de control y monitoreo del Robot Scorbot mediante señales Electroencefalográficas, se utilizó las señales eléctricas que proporciona la actividad cerebral, estas son procesadas por un software y con la ayuda de un protocolo de comunicación hace posible que gobierne a otras aplicaciones. [12].

1.3 Fundamentación Teórica

La implementación del sistema de control y monitoreo del robot Scorbot mediante las señales electroencefalográficas como se observa en la figura 1, se compone de diferentes elementos entre ellos se tiene un casco de señales electroencefalogramas que utiliza tecnología inalámbrica para su comunicación con su respectivo ordenador, además se utilizó un microordenador de bajo costo como servidor y es el encargo de proporcionar la información a sus clientes suscriptores, para realizar el sistema se requiere información que se detalla a continuación.

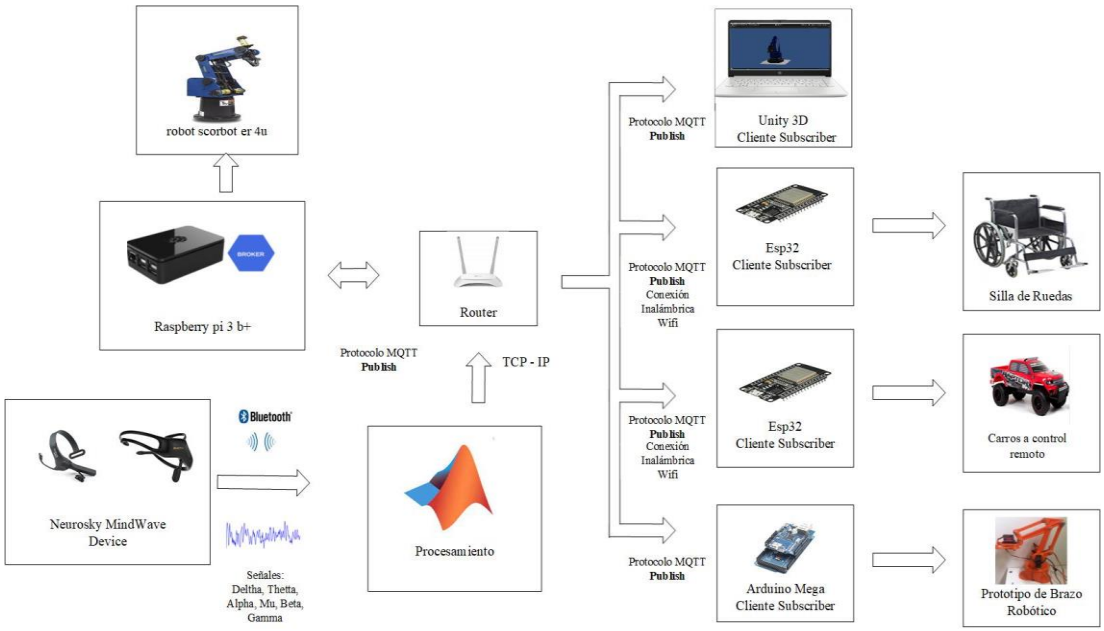


Figura 1: Esquema general del prototipo.

Elaborado por: El investigador

1.3.1 Electroencefalografía y Sistemas BCI

El electroencefalograma (EEG) puede detectar eficazmente las actividades fisiológicas del cerebro producidas principalmente por la corteza cerebral. El EEG detecta la diferencia de potencial obtenida por electrodos colocados en el cuero cabelludo, estas señales son generadas por la actividad neuronal en el cerebro, la amplitud de procesamiento de estas señales cubre el orden de microvoltios (μV). Del mismo modo, el potencial generado se debe a estímulos específicos, generalmente visual, auditivo,

etc, los canales EEG se clasifican según la frecuencia de la onda y cada canal relacionado con un estímulo específico[13].

Anatomía del encéfalo

El cerebro es un órgano que controla funciones como el razonamiento, las emociones y los movimientos corporales. Se divide en hemisferios izquierdo y derecho por la fisura longitudinal 2 y luego se divide en cuatro lóbulos en secuencia: lóbulo frontal, lóbulo parietal, lóbulo occipital y lóbulo temporal en la tabla 1 se detalla sus características. La corteza cerebral responsable de diferentes funciones se divide en varias áreas, como el lóbulo parietal responsable de la percepción, el lóbulo occipital responsable de la visión, el lóbulo parietal responsable de la audición y el lóbulo frontal control muscular y matemáticas para funciones como la resolución de problemas lógicos. La simetría entre los hemisferios no confiere la misma función, al contrario, tienen funciones diferentes. Esto significa que el hemisferio izquierdo controla la función de la parte derecha del cuerpo y la función del hemisferio derecho de la parte izquierda del cuerpo. En la figura 2, se detallan las partes de la corteza cerebral [14].

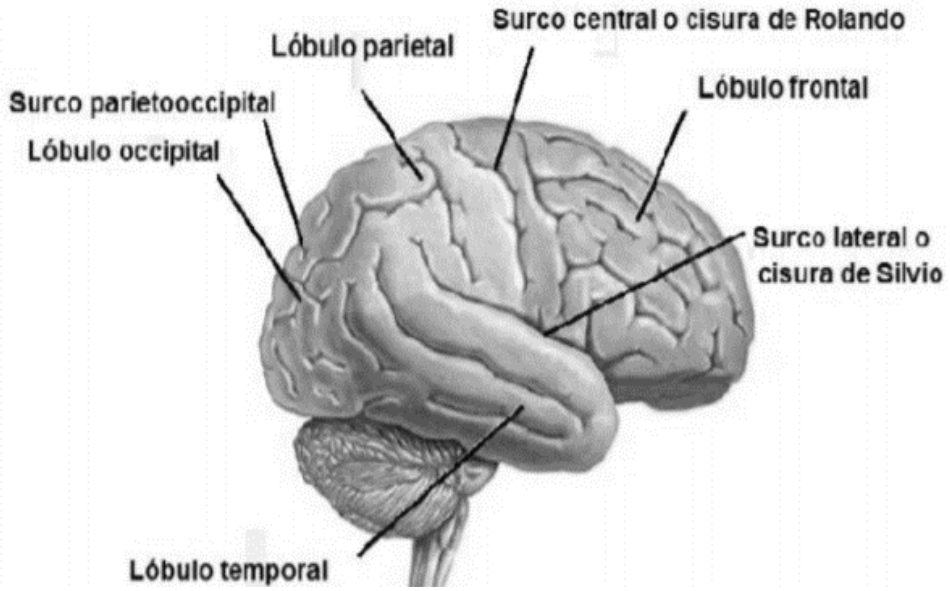


Figura 2: Áreas de la Corteza cerebral [14].

En la tabla 1, se encuentra conceptos relacionados al estudio del encéfalo y cerebro.

Tabla 1: Conceptos Ligados a la Anatomía del encéfalo [14].

Concepto	**Descripción**
Bulbo Raquídeo	Actúa sobre los movimientos involuntarios del corazón, interviene en el funcionamiento de las vías respiratorias.
Cerebelo	Integra el cerebro con la medula actuando de puente para la llegada y la transmisión de impulso entre ellos.
Cerebro	Es el centro de las funciones intelectuales, equilibra al organismo con el medio ambiente.
Diencéfalo	Se procesa las emociones mientras el tálamo recibe la información de diversos órganos sensoriales.
Telencéfalo	Se puede diferenciar dos hemisferios, izquierdo y derecho.
Hemisferio izquierdo	Procesa información de forma analítica
Hemisferio Derecho	Procesa la información de forma global, une y relaciona varios tipos de datos (sonido, imágenes, sentimientos y habilidades visuales.)
Corteza Cerebral o Córtex	Capa externa compuesta por una sustancia gris.
Lóbulo frontal	En él se lleva acabo “las funciones efectivas” que son las que asociamos a la toma de decisiones (uso de la memoria, planificación, selección de objetivos).
Lóbulo temporal	Es el encargado de la percepción, procesamiento y reconocimiento de los estímulos auditivos y olfativos.
Lóbulo occipital	Se ocupa de la percepción e interpretación de los estímulos visuales y del reconocimiento espacial.
Lóbulo ínsula	Es el centro de conexión entre el sistema límbico y el neocórtex, encargado razonamiento humano.

La Neurona

Las neuronas son células que tienen una gran capacidad para comunicarse con otras neuronas u otras células (ya sean nervios, glándulas o músculos) de forma precisa, rápida o incluso a grandes distancias. Estas son las encargadas de transmitir señales eléctricas, llamadas impulsos nerviosos para llevar a cabo esta comunicación de célula a célula. Los impulsos nerviosos se extenderán por toda la neurona y comenzarán a pasar por las dendritas hasta llegar al botón final, que eventualmente establecerá conexiones con otras neuronas, fibras musculares o glándulas. Ver figura 3 [14],[15].

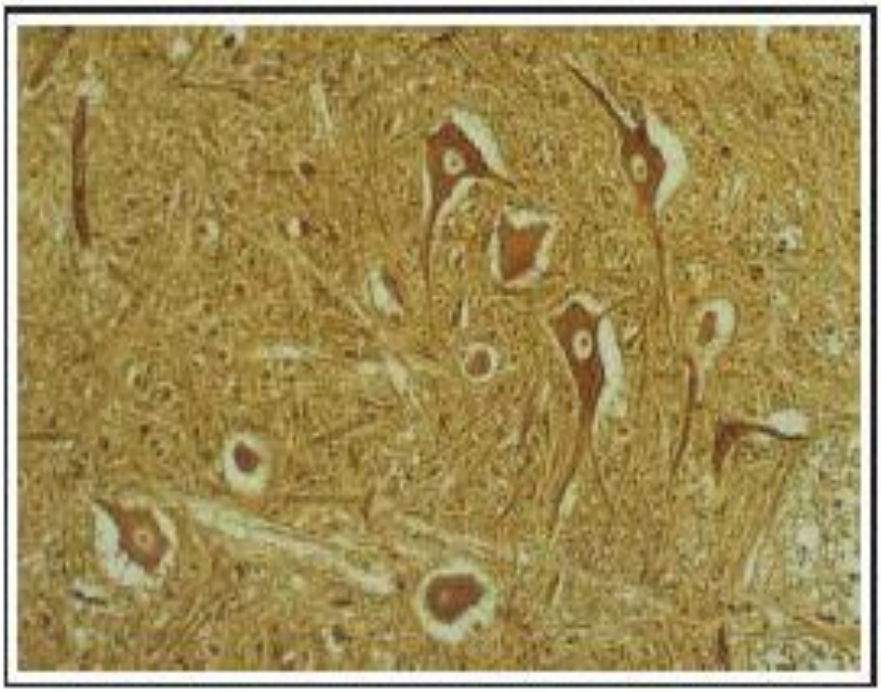


Figura 3: La neurona [14].

Cada neurona se compone de tres partes básicas: cuerpo celular, axón y dendritas. Cuerpo celular: También llamada célula somática, es la parte más gruesa que contiene componentes celulares como núcleo, ribosomas, retículo endoplásmico y mitocondrias en la figura 4 se observa la estructura de la neurona [14].

Axones: son lineales y transportan información electroquímica por toda la célula. Según el tipo de neurona, los axones pueden tener una capa de mielina que actúa como aislante. Este tipo de neurona se encuentra en nervios periféricos como el cerebro y la médula espinal, como sensores y motores, así como en neuronas no mielinizadas.

Células dendríticas: Son las encargadas de establecer contacto con otras neuronas para que puedan percibir el entorno externo, la longitud de una neurona puede variar desde unos pocos milímetros para las neuronas cerebrales hasta unos pocos decímetros para las neuronas táctiles [14],[15].

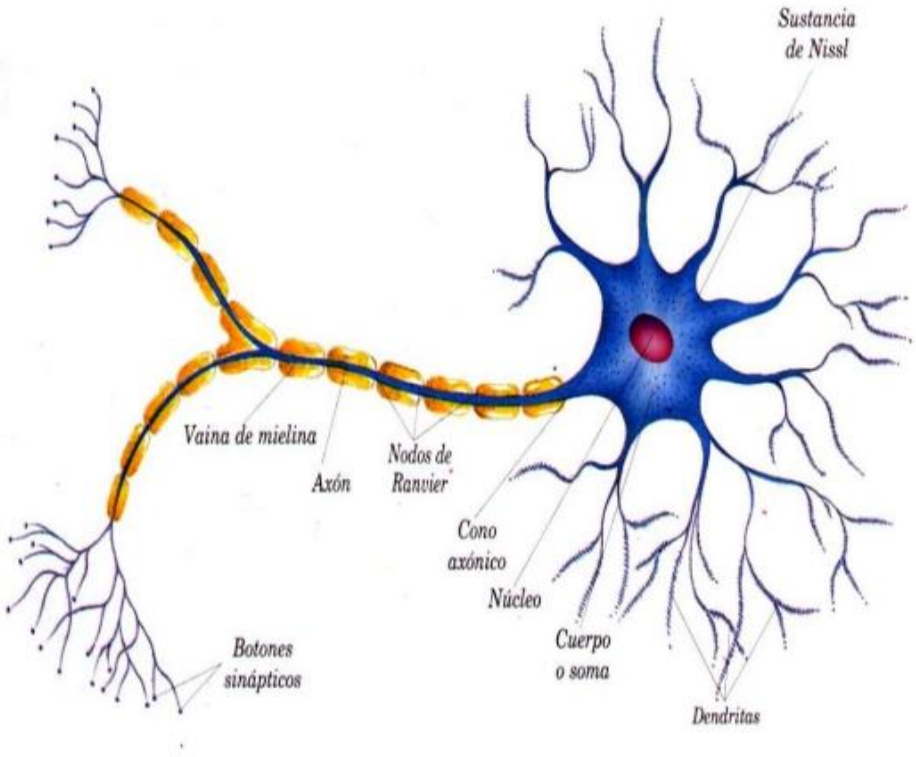


Figura 4: Topología de la neurona [15].

Transmisión eléctrica

Las ondas eléctricas transmitidas por las neuronas son causadas por cambios transitorios en la permeabilidad de la membrana plasmática. Su propagación se debe a una existencia de diferencia de potencial o potencial de membrana (debido a la concentración de iones es diferente entre la membrana interna y la membrana externa. Suele ser de –70 mV. el potencial eléctrico inactivo permanece en valor negativo (el interior con respecto al exterior) y cambios dentro de un margen estrecho. Cuando el potencial de membrana de las células excitables se despolariza más allá de cierto nivel umbral (de 65 mV a 55 mV), la célula genera un potencial de acción [14],[15],[16].

Entonces, la concentración de sodio interno provoca la descarga de iones. Restaurar el potasio y el sodio restaurando la carga negativa dentro de la célula. El potencial se vuelve más negativo, alcanzando -80 o -90 mV. El proceso toma alrededor de una milésima de segundo y después de un período de tiempo las células refractarias pueden repetir el proceso nuevamente. Esta actividad eléctrica se produce por estimulación del órgano del receptor, Oídos, ojos, tacto, etc., o también pueden estar relacionados con otros nervios. Todas estas actividades eléctricas se pueden medir para obtener un EEG [14],[15].

1.3.2 El electroencefalograma y los ritmos cerebrales

La actividad cerebral oscilatoria se produce en diversas zonas del cerebro y cambia sus características según el estado del sujeto, por ejemplo, quedarse dormido y estar despierto, o concentrarse en el trabajo o relajarse, la actividad oscilatoria en el EEG se divide en diferentes bandas de frecuencia o ritmos [17].

Las bandas de frecuencia que se pueden observar habitualmente son: Δ (1-4 Hz), θ (4-8 Hz), α y mu (8-13 Hz), β (14-30 Hz) y γ (30-40 Hz) como se puede apreciar en la tabla 2. Entre ellos, el ritmo de Mu mostró un gran interés. La particularidad es que al realizar o imaginar un movimiento, la amplitud de la oscilación disminuye. Los cambios en el ritmo mu se localizan en la corteza en correspondencia con los movimientos sensoriales de las partes del cuerpo a mover, algunos estudios han demostrado que se pueden distinguir los movimientos imaginarios de diferentes partes del cuerpo [17].

Los usuarios deben realizar un entrenamiento para controlar de una manera eficaz el sistema, mitigando posibles errores. Neuper y colaboradores realizaron un estudio en pacientes con debilidad motora severa para comprender si los pacientes pueden aprender a controlar su actividad cerebral. Los estudios han demostrado que la imaginación de los deportes producirá diferentes patrones de EEG oscilantes en el ritmo mu y el ritmo β del cuerpo de la persona [17].

Tabla 2: Tabla comparativa de señales cerebrales [17],[18].

Ondas	**Frecuencia (Hz)**	**Propiedades**	**Actividades Mental Relacionada**
Deltha (Δ)	< 4	Sección frontal adultos. Sección posterior en niños. Ondas de alta amplitud.	Ondas lentas relacionadas a la etapa de sueño profundo.
Theta (Θ)	4 - 7	Sección Frontal Media (Fz a Cz)	Inconciencia, meditación y somnolencia.
Alpha (α)	7 -12	Región posterior de la cabeza, ambos lados, amplitud alta en el lado dominante.	Relajación y concentración.
Mu (μ)	8 -13	Corteza sensorial-motora	Indicador de que las neuronas se encuentran trabajando.
Beta (β)	12 - 30	Corteza sensorial – motora, entre C3 y C4, distribución simétrica, más evidente en la parte frontal, ondas de amplitud baja.	Estado de alerta, pensativo y concentración activa.
Gamma (γ)	30	Corteza somato sensorial	Procesos somato-sensoriales. Mostrada durante cortos periodos, cuando la memoria reconoce objetos, sonidos o sensaciones táctiles.

Elaborado por: El investigador

Sistema BCI (Brain Computer Interface)

BCI se define como un sistema que puede medir y analizar señales cerebrales y convertirlas en salidas que no dependen de la ruta de salida normal de los nervios y músculos periféricos en tiempo real. De esta definición, se puede inferir que para que la operación BCI sea exitosa, debe haber un circuito de información cerrado entre los dos controladores adaptativos. El usuario que genera señales cerebrales específicas para codificar la intención, y convierte estas señales para cumplir con las condiciones.

La capacidad de comunicarse con los demás es una de las principales características del ser humano. A través de la comunicación, uno puede expresar pensamientos, deseos, sentimientos y desarrollo en la vida diaria. Las personas que están parcial o totalmente paralizadas debido a enfermedades como la esclerosis lateral amiotrófica, el infarto cerebral o la lesión de la médula espinal no tienen las habilidades de comunicación mencionadas anteriormente [17].

Para mejorar la calidad de vida de los pacientes con trastornos del movimiento, se están desarrollando sistemas innovadores en todo el mundo. Ellos llamaron Brain Computer Interface System o BCI (Brain Computer Interface, abreviatura en inglés). La idea principal es capturar las manifestaciones eléctricas, magnéticas o de otro tipo de la actividad cerebral y convertirlos en comandos que son interpretados y ejecutados por computadoras u otros dispositivos. Hoy en día, el sistema BCI se considera una herramienta con gran potencial que puede utilizarse para establecer alternativas de comunicación, restaurar las funciones y brindar rehabilitación a pacientes con trastornos neuromotores [17].

Componentes de un sistema BCI

En la figura 5 se muestra el principio fundamental de la interfaz cerebro ordenador y se describe de la siguiente manera: [17].

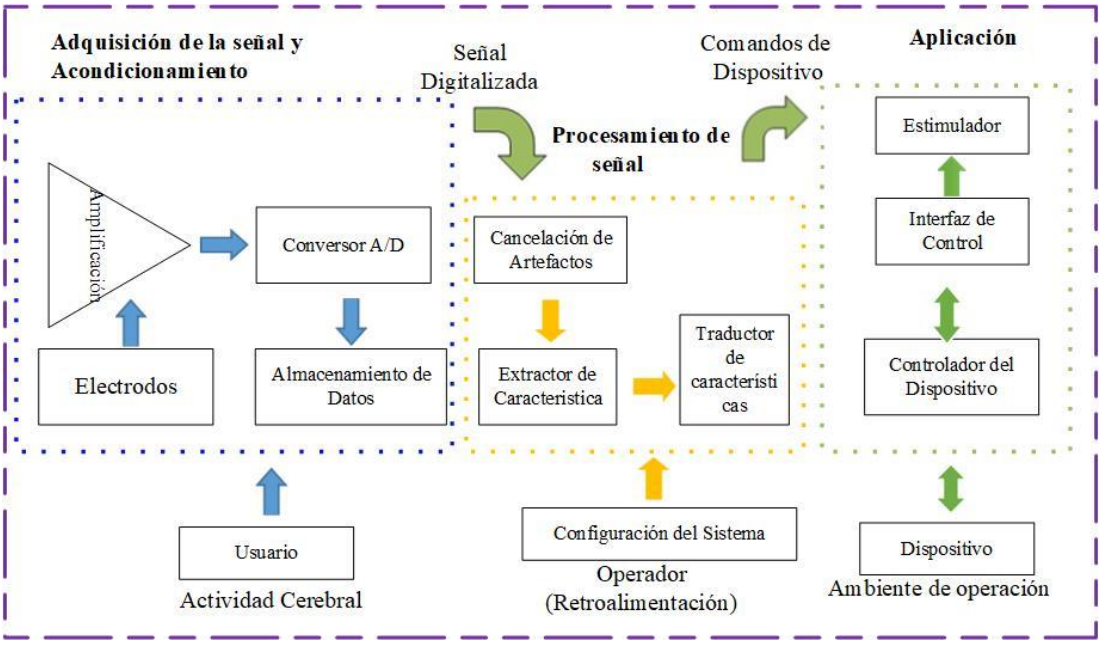


Figura 5: Diagrama de bloques de sistema BCI [17].

Elaborado por: El investigador

- **Adquisición y acondicionamiento de la señal**: Aquí se captura la actividad cerebral mediante electrodos colocados en la superficie de la corteza cerebral; y se acondiciona la señal mediante diferentes etapas de amplificación, filtrado y digitación. En esta etapa se puede incluir un dispositivo de almacenamiento de datos [17].
- **Procesamiento de la señal:** Dentro de esto se recibe la señal eléctrica cerebral digitaliza y se la transforma a comandos que entienda el dispositivo que el usuario quiera utilizar. Este bloque se divide en tres etapas que actúan de forma secuencial.
- **Cancelación de artefactos:** Se encarga de eliminar artefactos debidos otros tipos de actividades eléctricas producto del movimiento ocular y muscular o el producido por la línea eléctrica.
- **Extracción de características**: En esta etapa se traduce la señal cerebral de entrada en un conjunto de características correlacionado con el fenómeno asociado a la señal.
- **Traducción de características:** Se transforma el conjunto de características en una señal de control adecuada para el dispositivo que se pretender controlar.

- **Interfaz de control:** Recibe los comandos de control y realiza las acciones correspondientes en el dispositivo.
- **Estimulador:** Algunos sistemas incluyen un estimulador que es manejada por la interfaz de control. Señales de estimulación son enviadas al extractor de características para sincronizar la obtención de las mismas.
- **Configuración:** Permite a un operado definir y ajustar los parámetros del sistema que pueden ser definidos por el propio usuario.
- **Dispositivo:** Existen un rango limitado de dispositivos que pueden ser utilizados en un sistema BCI, como computadoras, sintetizadores de voz, neuroprotesis o encender o apagar la luz de la habitación.
- **Ambiente de operación:** Se refiere al ambiente físico como pared, piso, superficie, así como objetivos y personas en el ambiente afectan o puedan afectar en el funcionamiento del sistema [17].

1.3.3 Dispositivos de adquisición de señales encefalográficas

Casco Mindwave

El casco Mindwave es un dispositivo cuya función es medir la frecuencia de las ondas cerebrales, monitorizarlas y visualizarlas en la pantalla del ordenador, puede medir el nivel de relajación, meditación y concentración del usuario. Estos dispositivos no interfieren con la actividad cerebral, la diadema consiste en un sensor seco que ayuda a capturar la frecuencia de las ondas cerebrales, el sensor está ubicado en la frente, los neurocientíficos lo llaman área FP1. El dispositivo se comunica con la computadora a través de un dispositivo inalámbrico, a continuación, la Figura 6 muestra la estructura física de la diadema Neurosky [19].

Figura 6: Estructura de Mindwave Mobile [19].

Casco Emotiv EPOC headset

El dispositivo consta de 14 canales, estos canales se ubican específicamente en determinadas zonas del cuero cabelludo, sus dos puntos de referencia se ubican detrás de las orejas, y un receptor USB, a través del cual la señal adquirida se puede enviar a la computadora de forma inalámbrica, incluyen batería de litio recargable, en la figura 7 se observa los componentes del dispositivo para una mejor comprensión. El Auricular Epoc Emotiv también está equipado con un paquete de software de detección del estado emocional del usuario, las expresiones, y los comandos mentales entrenados por el usuario.

Los tipos de electrodos utilizados son superficiales y, debido a que son adherentes, se pueden distinguir. Esto significa que consisten en pequeños discos metálicos fijados con pegamento conductor, lo que proporciona una resistencia de contacto muy baja para la señal. Estos electrodos se fijan en una especie de brazo de plástico para asegurar su correcta posición porque pueden corregir ciertos puntos del cuero cabelludo [14].

Figura 7: Estructura de Emotiv EPOC headset [14].

Casco Emotiv Insight

Según el sistema internacional 10-20, Emotiv Insight tiene cinco sensores EEG ubicados en las siguientes posiciones: AF3, T7, PZ, T8 y AF4, como se muestra en la Figura 8. La resolución del casco es de 14 bits, 1 LSB = 0,51 µV. Además, el casco también cuenta con sensores de movimiento como giroscopio, magnetómetro y acelerómetro. Una característica importante del casco es su peso ligero, su señal fuerte y su forma fácil de usar [20].

Figura 8: Emotiv Insight [20].

1.3.4 Introducción a la robótica

Robótica

La robótica analiza y estudia una clasificación particular de sistemas mecánicos "robots manipuladores" estos sistemas por su estructura mecánica permite realizar una amplia variedad de aplicaciones, en dependencia de su utilidad cargando una reprogramación. La palabra mecatrónica proviene del idioma ingles: mecha que significa mecánico, y tronics que denota la parte electrónica; por tanto, integra la mecánica para la electrónica en una tarea específica. [21]

La robótica intenta implementar sistemas autómatas a un nivel lógico, con razonamiento e inteligencia, que puedan desempeñar labores de los seres humanos; tareas específicas que requieren precisión y gran destreza, todo este con la finalidad de implantar dispositivos automáticos o de control a distancia los mismos que realicen trabajos de alto riesgo o a su vez imposibles para los seres humanos y con ello remplazar parcial o totalmente algunas labores realizadas por los humanos. [21]

En el artículo titulado "Rumaround" por Asimov de 1952, se postulan tres leyes de la robótica que siguen vigentes hasta la actualidad las mismas que son:
Leyes de la robótica: [21]

- Un robot no debe dañar a un ser humano, ni dejar que el ser humano sufra daño.
- Un robot debe obedecer las órdenes que le son dadas por un ser humano, excepto cuando estas órdenes estén en oposición a la primera ley.

- Un robot debe proteger su propia existencia, hasta el límite en que esta protección no entre en conflicto ni con la primera, ni la segunda ley

Cuando la aportación literaria de Asimov llegó a su momento de auge este consideró necesario añadir una cuarta ley, antepuesta a todas las demás en la que afirma que un robot no debe actuar solo para complacer las necesidades individuales, sino más bien su accionar debe preservar el beneficio común de toda la humanidad [21].

Robot

Un robot es un manipulador automático multifuncional reprogramable, capaz de posicionar y orientar piezas, herramientas o equipos especiales según trayectorias variables reprogramables para realizar diversas tareas. Suele aparecer en forma de uno o más brazos que terminan en una muñeca. En la tabla 3 se detalla conceptos fundamentales del robot. Su unidad de control incluye un dispositivo de almacenamiento. Habitualmente, su finalidad es realizar tareas cíclicas para que pueda adaptarse a otra tarea sin cambiar permanentemente sus materiales [22].

El robot recibe pedidos u órdenes de forma automática, y se utiliza para complementar actividades complejas de trabajo, mueve objetos en su entorno físico. Tenemos cuatro características principales: autonomía, interactividad, movilidad y comunicación Además, el término robot también abarca áreas relacionadas con los sistemas autónomos, como la robótica, la visión y la inteligencia artificial [22].

El uso de robots puede mejorar la calidad de los productos terminados y reducir los riesgos laborales, reduciendo así los costos de producción. Varios países del mundo han invertido mucho en el uso de nuevas tecnologías de producción y Estados Unidos está invirtiendo cada vez más en nuevas formas de automatización para aumentar la productividad y reducir los costos de fabricación [22].

Tabla 3: Conceptos Ligados a los robots [21],[22] .

Concepto	Descripción
Manipulador	Equipo electromecánico capaz de interactuar con su entorno.
Efector Final	Herramientas, pinzas u otros equipos instalados en el extremo del manipulador para realizar tareas útiles.
Espacio de Trabajo	El espacio donde el efector final del robot puede alcanzar la posición y dirección.
Posición	La posición de traslación del objeto (línea recta).
Orientación	La posición de rotación (ángulo) del objeto.
Articulación	Un dispositivo que permite el movimiento relativo entre dos enlaces en un robot.
Cinemática	El estudio del movimiento sin tener en cuenta las fuerzas que lo provocan.
Dinámica	El estudio del movimiento con respecto a las fuerzas que lo producen.
Actuador	Proporciona fuerza para el movimiento del robot.
Sensor	Lee variables reales en el movimiento del robot para usar en el control.

SCORBOT ER 4U

El robot SCORBOT-ER 4u es un robot industrial, una máquina general programable con algunas características antropomórficas. El brazo robótico y las capacidades de programación del robot lo hacen muy adecuado para diversas tareas de producción. El robot SCORBOT-ER 4u es un sistema versátil y confiable para fines educativos. El brazo robótico se puede montar en una mesa o base deslizante lineal. La velocidad y la repetibilidad del robot lo hacen ideal para el funcionamiento autónomo y también adecuado para la integración en aplicaciones de celdas de trabajo automatizadas como soldadura robótica, visión artificial, colocación de máquinas CNC y otras operaciones[23].

El brazo robótico está conectado al controlador mediante un cable que se conecta desde la parte inferior del brazo al conector D50. El cable del motor está conectado directamente al conector D50. Estos cables son extremadamente flexibles y no se romperán incluso después de que el brazo de Scorbot se mueva. Se detalla las especificaciones del robot scorbot en la tabla 4. [23].

Tabla 4: Especificaciones técnicas Scorbot ER 4U[23].

SCORBOT -ER 4U

	Concepto	**Descripción**
1	Estructura mecánica	Articulado Vertical
2	Numero de ejes	5 ejes
3	Eje Movimientos	310
4	Eje 1: Giro Básico	158
	Eje 2: Rotación del hombro	260
	Eje 3: Rotación del codo	260
	Eje 4: Paso de la muñeca	(mecánica) + - 570
	Eje 5: Rollo de muñeca	(eléctricamente)
5	Operación del radio máximo	610 mm
6	Realimentación	Codificador óptico en cada eje
7	Actuadores	Motor 12 VDC servo

Elaborado por: El investigador

1.3.5 Tarjetas microcontroladoras de bajo costo

Raspberry

La Raspberry Pi es una computadora de bajo costo y con un tamaño compacto, puede ser conectada a un monitor de computador o un TV, y usarse con un mouse y teclado estándar. Es un pequeño computador con un sistema operativo linux capaz de permitirle a las personas de todas las edades explorar la computación y aprender a programar lenguajes como Scratch y Python [24].

Raspberry Pi 3 B + tiene un GPIO de 40 pines, lo que permite que los sensores y actuadores se pongan en contacto con el mundo exterior. Es importante saber que el voltaje de trabajo de Raspberry GPIO es de 3.3V, como se muestra el figura 9 [24].

Figura 9: Tarjeta Raspberry Pi 3 B+ [24].

Nvidia Jetson Nano

Es una computadora pequeña y poderosa que le permite ejecutar múltiples redes neuronales en paralelo para aplicaciones como clasificación de imágenes, detección de objetos, segmentación y procesamiento de voz. Plataforma fácil de usar con un consumo de energía tan bajo como 5 vatios. NVIDIA Jetson Nano es un sistema en módulo (SoM) y kit de desarrollo de la serie NVIDIA Jetson, que incluye una GPU Maxwell integrada de 128 núcleos, CPU ARM A57 de 64 bits, memoria LPDDR4 de 4 GB y soporte. Para E / S de alta velocidad MIPI CSI-2 y PCIe Gen2, como se muestra en la Figura 10 [25].

Figura 10: Placa física Nvidia Jetson Nano [25].

Jaguar One

Jaguar Electronics Hong Kong Limited lanzó oficialmente su primera computadora de escritorio Jaguarboard exclusivo basado en X86 basado en procesadores Intel Atom de cuatro núcleos, que proporciona 1 GB de RAM DDR3L y 16 GB El módulo de

memoria flash eMMC simplifica enormemente herramientas de desarrollo, también permite a aquellos que aún no están familiarizados con la arquitectura ARM sin perder tiempo y esfuerzo aprenda nuevos conocimientos antes de poner Jaguarboard en el proyecto. Es ampliamente compatible con las distribuciones principales de Linux, como Ubuntu, Debian y Red Hat y versiones estándar de Windows 8.1 / 10 y Android, en la figura 11 se muestra su estructura física [26].

Figura 11: Placa física Jaguar One [26].

1.3.6 Protocolos de Comunicación en la Industria

Con la aparición de los dispositivos IoT, surge el concepto de industria 4.0. Podemos definirla como la digitalización completa a través de la integración de tecnologías de procesamiento de datos, software inteligente y sensores, desde los proveedores hasta los clientes. Con el fin de esta discusión, es importante agrupar los protocolos en dos categorías: cliente/servidor (client/server) y publicar/suscribir (publih/subscribe)[27] .

Los protocolos cliente/servidor requieren que el cliente se conecte al servidor y realice solicitudes. Por otro lado, los protocolos publicar/suscribir requieren que los dispositivos se conecten a un “tópico” de un gestor intermediario y publiquen la información [27].

Existen varios protocolos: algunos son privados y otros que son estándares abiertos, en los estándares abiertos tenemos protocolos que tienen el potencial de conectar dispositivos industriales con diferentes plataformas. Pero para que esto sea posible, los

dispositivos deben poder comunicarse, tanto entre sí, como hacía el exterior. Por ello comentamos algunos de los protocolos de comunicación existentes en la industria [27].

UDP

El protocolo UDP (Protocolo de datagramas de usuario) es un protocolo que permite la transmisión sin conexión de datagramas en redes basadas en IP. Para obtener los servicios deseados en los hosts de destino, se basa en los puertos que están listados como uno de los campos principales en la cabecera UDP. Como muchos otros protocolos de red, UDP pertenece a la familia de protocolos de Internet, por lo que debe clasificarse en el nivel de transporte y, en consecuencia, se encuentra en una capa intermedia entre la capa de red y la capa de aplicación
[27].

Cada envío de datos corresponde a un único envío de un datagrama independiente del resto de datagramas y de la misma comunicación, siguiendo los preceptos de encaminamiento de datagramas por Internet, la entrega al destino no está asegurada por el propio protocolo. Debido a que UDP utiliza IP para su transporte por Internet, en caso de ser, este se fragmentará y ninguno de estos fragmentos, proporcionara ningún tipo de seguridad o fiabilidad en la entrega, debido a la sencillez de este protocolo, el diseño de su cabecera, es mucho más simple que el de IP [27].

TCP/IP

La sigla TCP/IP significa Protocolo de control de transmisión/Protocolo de Internet. Proviene de los nombres de dos protocolos importantes incluidos en el conjunto TCP/IP, es decir, del protocolo TCP y del protocolo IP se podría definir como un protocolo orientado a una conexión fiable y orientada a un flujo de bytes. Aunque el protocolo TCP al igual que UDP utiliza los servicios de IP para su transporte por Internet, es un protocolo orientado a conexión. Esto significa que las dos aplicaciones envueltas en la comunicación (usualmente un cliente y un servidor), deben establecer previamente una comunicación antes de poder intercambiar dato [28],[27].

HTTP

El HTTP las siglas que en inglés significan Hypertext Transfer Protocol, cuyo equivalente en nuestro idioma sería el de Protocolo de Transferencia de Hipertexto, es un protocolo cliente/servidor de la Capa de Transporte y los datos pueden presentarse en múltiples formatos tales como; HTML, JavaScript, JavaScript (JSON), XML, etc. De todos estos formatos el más usado, para el intercambio de información entre dispositivos, es JSON pues es ligero y flexible y disponemos de infinidad de librerías para usarlo con cualquier lenguaje de programación [27].

HTTP está pensado para que los nodos clientes puedan tener acceso a los recursos del nodo servidor a través de solicitudes, donde en la mayoría de los casos, un recurso es un dispositivo. Aunque HTTP es ampliamente usado en la industria (sobre todo en su versión segura HTTPs), básicamente se utiliza para configuración de dispositivos, pero no para acceso a los datos que éstos contienen. También existe el problema de la interoperabilidad ya que aunque multitud de dispositivos soportan HTTP (REST/JSON) no son totalmente compatibles entre ellos y normalmente debe realizarse algún ajuste para que puedan comunicarse [27], [29].

MQTT

MQTT son las siglas MQ Telemetry Transport, aunque en primer lugar fue conocido como Message Queing Telemetry Transport. Es un protocolo de comunicación M2M (machine-to-machine) de tipo message queue, fue creado por técnicos de IBM y Arcom en 1999 como un modo rentable y seguro de supervisar dispositivos de medición remotos. Usa un modelo de comunicación muchos con muchos a través de un servidor centralizado llamado Broker [30].

Este protocolo que funciona sobre TCP (aunque existe una versión denominada MQTT-S que lo hace sobre UDP), pertenece a los denominados de publicación/suscripción y ha comenzado a utilizarse a gran escala tras la aparición del Internet de Cosas (IoT) pues como veremos dispone muy buenas cualidades para usarse en este campo [30].

MQTT es un protocolo abierto, sencillo, muy ligero y muy fácil de implantar. Los dispositivos publican información sobre un asunto o tópico en el broker quien la reenvía a su vez a aquellos clientes que se hayan suscrito al mismo, en la figura 12 se ilustra el método de trabajo de mqtt [30],[31].

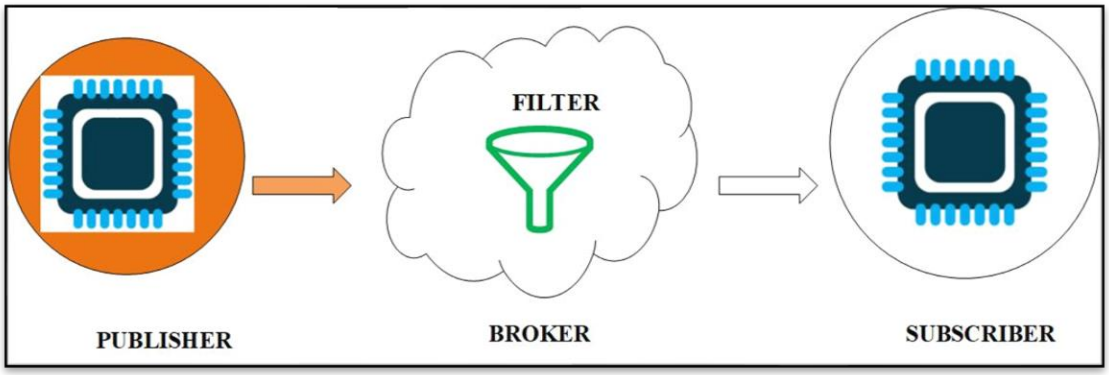


Figura 12: Estructura MQTT [26].

Elaborado por: El investigador

MQTT se considera el mejor protocolo para datos en "tiempo real", porque el intermediario separa al editor del suscriptor, ahorrando así tiempo de procesamiento y energía. Sus ventajas incluyen:

- Especialmente adecuado para utilizar el ancho de banda más pequeño
- Ideal para usar redes inalámbricas
- Consume muy poca energía
- Muy rápido y el tiempo de respuesta es mayor que el de otros protocolos web actuales.

1.4 Objetivos

Objetivo general

Implementar un sistema de manipulación y monitoreo del robot Scorbot mediante las señales electroencefalográficas (EEG)

Objetivos Específicos

- Analizar los sistemas de control inteligente utilizando cascos de señales electroencefalográficas (EEG).

- Determinar las tecnologías de software y hardware necesarios para el desarrollo del sistema de manipulación y monitoreo del robot scorbot.

- Diseñar el sistema de manipulación y monitoreo del robot scorbot mediante señales electroencefalograficas (EEG)

El objetivo del presente proyecto de investigación se enfoca en el desarrollo de un sistema de manipulación y monitoreo del robot scorbot mediante señales electroencefalográficas (EEG) que aporta de manera significativa a las industrias transfiriendo el estado mental en una forma de dominio, además de controlar otras aplicaciones ayudando a usuarios con discapacidad motoras. Para lograr dicho objetivo será necesario realizar las actividades detalladas en cada objetivo.

Para el análisis de los sistemas de control inteligente utilizando cascos de señales electroencefalográficas (EEG), el presente proyecto parte de una búsqueda de información acertada y verificada por fuentes bibliográficas confiables a cerca de los sistemas de control ocupados a nivel global. Para la consecuencia de este objetivo será necesario la realización de las siguientes actividades:

1. Investigar los métodos de adquisición de las señales electroencefalográficas y los sistemas de control existentes.
2. Analizar las señales electroencefalográficas.

Para obtener resultados favorables en la implementación hay que escoger de manera adecuada el software y hardware, en vista de que existen una gama amplia en el mercado tecnológico, para lo cual se realizarán las siguientes actividades:

1. Determinar el esquema general del sistema, con sus respectivas características.
2. Comparar los equipos y dispositivos existentes en el mercado que se acoplen de mejor manera a los requerimientos del sistema.

Una vez adquiridos los equipos y en base a la información obtenida, se elaborará el diseño y posteriormente la implementación del sistema de manipulación y monitoreo del robot scorbot mediante las señales electroencefalográficas (EEG), cumpliendo con los requerimientos exigidos para realizar la manipulación y monitoreo del robot scorbot, para ello se establecieron las siguientes actividades:

1. Desarrollar cada uno de las etapas que componen el sistema con sus respectivos compontes de acuerdo con el esquema general.
2. Realizar pruebas de funcionamiento y corrección de errores en el sistema.

CAPÍTULO II.
METODOLOGÍA

2.1 Materiales

El desarrollo de la metodología para el presente proyecto se requiere de los siguientes materiales: artículos publicados en revistas científicas, proyectos desarrollados de investigación afines al tema principal, libros, información sobre sistemas que utilizan tecnología idéntica al presente proyecto, dispositivos y equipos, en lo que corresponde al Hardware del sistema, por otro lado en el software se utilizó aplicaciones para la configuración de los mismos equipos como también para programación del desarrollo del sistema

2.2 Métodos

2.2.1 Modalidad de Investigación

En el presente proyecto de investigación se realizó de acuerdo a los conceptos de la investigación aplicada ya que está directamente relacionado con la solución a los problemas de personas con discapacidad motora, además se tiene por objetivo la generación de conocimiento que será utilizado en la manipulación del robot Scorbot por medio de los siguientes tipos de investigación:

Se aplicó la investigación bibliográfica – documental en libros, tesis, artículos científicos etc. Para la adquisición de información relevante acerca del tema de estudio, igualmente como proyectos realizados con anterioridad verificando los resultados y problemas obtenidos por otros investigadores.

El robot Scorbot – ER 4U está ubicado en la Facultad de Ingeniería en Sistemas, Electrónica e Industrial de la Universidad Técnica de Ambato por esa razón se utilizó la investigación de campo

2.2.2 Recolección de Información

Para la recolección de información se optó por el uso de documentos, revistas, libros, proyectos desarrollados, garantizando la información evidente e íntegro. Se tomó información de las pruebas realizados, tomando tiempos de retardo en la comunicación entre el casco y el manipulador.

2.2.3 Procesamiento y Análisis de Datos

Para el procesamiento y análisis de los datos se procedió con las siguientes actividades: El procesamiento y análisis de datos se realizó mediante una clasificación de la documentación obtenida, presentando una descripción ordenada de los entornos a estudiarse en el proyecto. Se realizó un análisis crítico de los datos obtenidos durante la recolección de información, considerando los siguientes lineamientos:

- Análisis y organización de toda la información recolectada.
- Obtener parámetros técnicos, específicos y concretos que determinen las características del sistema a ser diseñado.
- Interpretar la información que permite plantear estrategias de solución al problema.
- Planteamiento de la propuesta solución.

2.2.4 Desarrollo del Proyecto

A continuación, se presenta el desarrollo de las actividades necesarias que se efectuaron para el desarrollo del proyecto:

- Análisis sobre sistemas de control inteligente para robots industriales mediante las señales electroencefalográficas (EEG).
- Recolección de información sobre las etapas de adquisición de señales electroencefalográficas.
- Análisis de la información existente para el filtrado y separación de los tipos de señales encefalografías.
- Investigación de las características de los dispositivos para el diseño del prototipo.
- Selección de tecnologías de procesamiento y sensorización para señales electroencefalográficas.
- Desarrollo de la etapa de adquisición de señales EEG.
- Selección de software y hardware para la manipulación y monitoreo del robot scorbot.
- Desarrollo de la etapa de manipulación y monitoreo del robot scorbot
- Construcción del prototipo

- Realización de pruebas de funcionamiento del prototipo mediante las ondas cerebrales.
- Corrección de errores y validación del prototipo.
- Elaboración de informe final

CAPÍTULO III

RESULTADOS Y DISCUSIÓN

3.1 Análisis y discusión de los resultados

3.2 Desarrollo de la propuesta

En el presente trabajo se desarrolló un sistema de manipulación y monitoreo del brazo robótico industrial Scorbot, desde una interfaz gráfica desarrollada en Matlab con la utilización de los cascos de señales electroencefalográficas, el controlador del brazo es la tarjeta SBC raspberry pi la cual interactúa con el manipulador robótico a través de los pines de entrada / salida GPIO. Para la comunicación entre interfaz gráfica y el controlador se utilizó el protocolo de comunicación MQTT, pero también por interés de demostrar que el sistema funciona para cualquier tipo de aplicación se probó en una silla de ruedas, un carro a control remoto, un prototipo de brazo robótico y además de simulación en unity 3D de un brazo robótico.

3.2.1 Requerimientos para el desarrollo del sistema

Para el desarrollo del sistema de manipulación y monitoreo del robot scorbot mediante las señales electroencefalográficas se plantearon algunos requerimientos:

- Adquisición de las señales electroencefalográficas
- Procesamiento y monitoreo de las señales Electroencefalográficas
- Accionamiento de las articulaciones del robot y los actuadores de las diferentes aplicaciones.

3.2.2 Diagrama general del sistema

En la figura 13 se observa el diagrama de bloques del sistema el cual se encuentra divido 3 etapas. La primera consiste en la adquisición de las señales eléctricas producidas por la actividad del cerebro mediante los dispositivos auriculares neuroskyn y emotiv insight, que gracias a la tecnología inalámbrica se enlaza a la segunda etapa esto con el fin de procesar, filtrar, visualizar las señales mediante algoritmos, la tercera etapa cuenta con la recepción de los datos y el accionamiento de las articulaciones del brazo como también el accionamiento de los actuadores de las aplicaciones.

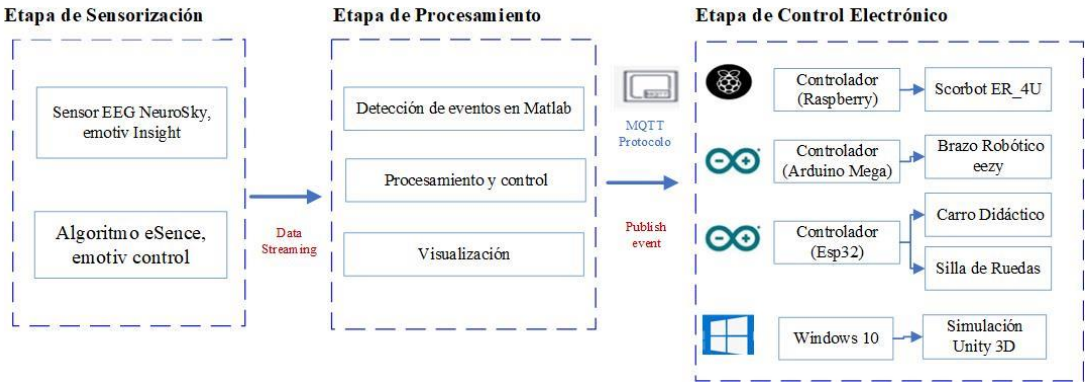


Figura 13: Esquema General del Sistema.

Elaborado por: El investigador

3.2.3 Selección de elementos para la implementación del sistema

Los elementos que forman parte del sistema se han seleccionado acorde al diagrama de bloques presentado en la figura 13, para lo cual, se realizaron un análisis técnico de cada uno de los componentes y dispositivos que intervienen, y de esta manera, seleccionar la mejor opción para la implementación del sistema en cada una de sus etapas, en el esquema general se encuentra conformado por los siguientes módulos:

- Sensorización
- Control y procesamiento
- Comunicación
- Interfaz de control electrónico

Se procede en la selección y descripción de los mismo, según las características para el correcto funcionamiento del sistema los cuales son los siguientes:

Sensorización

Cascos de señales electroencefalográficas (EEG)

Es el encargado de la adquisición de las señales electroencefalográficas captada por los electrodos, es necesario pensar en la etapa de sensado; que se encarga de recolectar los datos con el sensor ubicado en la parte frontal de la cabeza (frente) del usuario, este es el primer paso para obtener las señales EEG. Para la adquisición de las actividades cerebrales se requiere un electroencefalograma (EEG). En la tabla 5 se describe los cascos EEG más utilizados en aplicaciones BCI, se describe sus características:

Tabla 5: Casco de señales EEG[20].

Características	Mouse	Mindawave	Emotiv Insight	Emotiv Epoc
Descripción				
Frecuencia	220Hz	512Hz	256Hz	256Hz
Electrodos	Mojados	Secos	Mojados	Mojados
Conectividad		Conexión Directa al electro de seco	Bluetooth 4.0 LE. Wireless Banda de 2.4GHz	Bluetooth Smart. Banda: 2.4GHz
Batería		8 horas AAA. Interna de 15 mA a 3.3 V	Interna del polímetro de litio 480 mAh.	Interna del polímetro de litio 640 mAh.
Canales	2	1	5	14
Tiempo de configuración	5	3	15	15
Precio	$ 30.00	$ 80	$ 530	$ 799

Elaborado por: El investigador

Una vez realizado el análisis comparativo se seleccionó la diadema Mindwave de NeuroSky y emotiv insight ya que ofrecen las tecnologías de sensor de ondas Cerebrales más económicas combinadas con los conjuntos de herramientas más accesibles y abiertos que serán útiles a la hora de implementar el prototipo; así como su tiempo de configuración.

Control y procesamiento

Ordenador de Placa reducida (SBC)

La información proveniente de los auriculares de señales electroencefalográficas, una vez procesados se transmite por medio de tecnología wifi hacia servidor MQTT, con el fin que estos datos se han transmitidas a sus clientes, además es el controlador del brazo que interactúa con el manipulador robótico a través de los pines en entrada y salida GPIO, para ello se utiliza Rasberry Pi. En la tabla 6 se visualiza la selección de la mejor alterativa de un SBC analizando las características de diferentes tipos.

Tabla 6: Ordenador de Placa reducida

Características	**Raspberry Pi 3 B+**	**Nvidia Jetson Nano**	**Jaguar One**	**LattlePanda**
Descripción				
Procesador	1.4 GHz 64-bit Quad-Core ARMv8	A57 @ 1.43 GHz	1.3 GHz Quad-Core Intel Atom Z3735G/F	1.8 GHz 64-bit Quad-Core Intel Z8300
Memoria de almacenamiento	Tarjeta microSD	LPDDR4 25.6 GB / s	16gb 16gb	16gb 16gb
Memoria RAM	2 gb	4 gb de 64 bits	1gb	2 y 4 gb
Conectividad	4 USB 2.0 HDMI 1.4 Wifi Bluetooth 4.1 Ethernet	Gigabit Ethernet, M.2 Key E	3 USB 2.0 HDMI 1.4 Ethernet	1 USB 3.0 2 USB 2.0 Wifi Bluetooth 4.0 Ethernet
Sistema Operativo	Linux Windows	Linux Windows	Linux Android Windows	Windows
Valor $	80	350	80	160

Elaborado por: El investigador

Se optó como ordenador para el presente proyecto la Raspberry Pi modelo PI3 B+, por las características que cuenta y puede aportar al mismo, entre los beneficios más importantes destaca el procesador que funciona a 1.4 Ghz, alcanzando una confianza de no tener complicaciones al correr aplicaciones que necesita el proyecto dentro de este dispositivo, un punto importante es la tarjeta de red, el dispositivo cuenta con Gigabit Ethernet, que es capaz de alcanzar los 300 Mbps al funcionar sobre USB 2.0, añadiendo su bajo costo y características técnicas excelentes, ante los demás competidores.

Placa Electrónica

Para el accionamiento de las diferentes aplicaciones como: la silla de ruedas, el carro a control remoto y el prototipo de brazo robótico, para estas actividades se requiere de una placa electrónica de acuerdo a las características de memoria, pines de entrada y salida analógica como digitales, conexión inalámbrica y disponibilidad en el mercado, por tal motivo se analizó las tarjetas como Arduino y NodeMCU que se detalla en la Tabla 7.

Tabla 7: Placas Electrónicas

Placa Características	**Arduino Mega 2560**	**NodeMCU ESP8266**	**Esp32**
Descripción			
Voltaje de entrada	5-12V	5-12 V	5-12 V
Frecuencia de reloj	48 MHz	80MHz – 160Mhz	160MHz – 240MHz
Pines digitales I/O	54	11	34
Corriente de operación	40 mA	80 mA	80 mA
Conexión Inalámbrica	No	Wi-Fi	Wi-Fi
Procesador	32 bits ARM	Tensilica Xtensa LX3 de 32 bits	Tensilica Xtensa LX3 de 32 bits Dual Core
Memoria Flash	256Kb	256Kb	32Mb
Comunicación	I2C/SPI Serial	I2C/SPI Wifi	I2C/SPI Wifi
Costo	**$ 15.00**	**$ 8.00**	**$ 15.00**

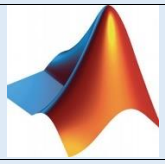
Elaborado por: El investigador

En la Tabla 7 se realizó la comparación de los microcontroladores más utilizados y disponibles en el mercado, además de sus características similares se decidió en la utilización de la Esp32, ya que permite la comunicación inalámbrica y así evitar la utilización de módulos adicionales como Wi-Fi o Bluetooth y para la manipulación del brazo robótico se decidió por la tarjeta controladora Arduino mega, cumpliendo con los requerimientos para el sistema de control, además el costo y tamaño son accesibles para la realización del proyecto.

Software de procesamiento

El software para el análisis de las señales obtenidas por los cascos electroencefalográficos, deben tener varias características como la capacidad de evaluación de los datos, la rapidez de procesamiento, programación, almacenamiento, entre otros. En la tabla 8 se describen los softwares más utilizados en el procesamiento de las señales electroencefalográficas con sus características:

Tabla 8: Software de procesamientos de señales

Características	**Labview**	**Scilab**	**Matlab**
Descripción			
Sintaxis	La sintaxis de scripts de archivos .m con el MathScript Node	Lenguaje orientado al uso de matrices y vectores.	Lenguaje Propio, puede ejecutarse tanto entorno interactivo, como en archivos scripts.
Tipos de análisis	Herramienta grafica y textual proceso digital de señales y manejo de gráficos dinámicos.	Está diseñado para el tratamiento de polinomios y calculo simbólico.	Operación con matrices, funciones, algoritmos y creación de interfaces de usuario (GUI) y la comunicación.
Plataformas	Unix, Mac y GNU/linux	Windows, Mac y Linux	Windows, GNU/Linux y Mac OSx
Ventajas	El sistema está dotado de un compilador grafico para lograr la máxima velocidad de ejecución posible.	Plataforma de programación a nivel superior permite todo tipo de análisis que se desee efectuar.	Plataforma programación, modelado de y simulación basada en la nube, análisis de datos, crear modelos y ejecutar.

Elaborado por: El investigador

De acuerdo con los requerimientos del sistema, después del análisis detallado en la tabla anterior se seleccionó al software Matlab ya que cubre en la actualidad prácticamente casi todas las áreas principales de ingeniería y la simulación, destacando entre ellos el procesamiento de señales, imágenes, control robusto, redes neuronales, lógica difusa, etc. También la fácil integrar hardware de cualquier proveedor,

desarrollar algoritmos de análisis de datos y diseñar interfaces de usuario personalizadas, gracias al entorno de programación visual disponible para realizar y ejecutar programas que necesiten ingreso continuo de datos.

Software Emotiv Xavier

La empresa Emotiv provee del software de entrenamiento e interpretación de datos, con la finalidad de expresar información útil, accesible por el usuario. En la figura 14 se observa la interfaz que maneja el software Xavier, permitiendo adquirir seis medidas del estado mental. En tabla 9 se visualiza la descripción de las seis medidas de estado mental.

Figura 14: Interfaces del Emotiv

Elaborado por: El investigador

Tabla 9: Descripción de las medidas de estado mental

Señales	Descripción
Interés 	Es el grado de atracción a la actividad actual que realiza el usuario. El interés está relacionado con el disfrute de la tarea actual
Excitación 	Captura el nivel de entusiasmo emocional, el cual es un estado de mayor actividad, tanto mental como física, lo cual incrementa el nivel de alerta
Compromiso 	Mide que tan inmerso se encuentra el usuario en las tareas que se encuentra experimentando. El compromiso requiere conjuntamente los procesos de atención y concentración.
Enfoque 	Es una medida de la atención fija a una tarea específica. Mide la profundidad de la atención, así como la frecuencia que su atención cambia entre tareas. Un alto nivel de conmutación de tareas es una indicación de bajo enfoque y distracción
Estrés 	Es una medida del nivel de comodidad con el reto actual. Los altos niveles de estrés pueden derivarse de una incapacidad para completar tareas difíciles; lo cual hace que la persona se sienta abrumada y genere temor ante las posibles consecuencias negativas.
Relajación 	Es la capacidad de alcanzar un estado mental tranquilo. Es una medida de la habilidad para reducir los niveles de activación, permitirse descansar y recuperarse de una concentración intensa.

Elaborado por: El investigador

Comandos Mentales del Emotiv

En la figura 15 se aprecia la interfaz de comandos mentales, las acciones que se pueden realizar por el usuario son 13 e incluye 6 direccionales y 6 movimientos de rotación de un cubo 3D. Hay una fase de tren de aprendizaje supervisado donde el operador se ve obligado a realizar una variedad de pruebas repetitivas mientras piensa en una cierta acción. Las acciones varían al concentrarse en moverse a la izquierda, a la derecha, hacia delante, o hacia arriba y etc., y cuando se realiza cierta acción se visualiza en un juego con el cubo.

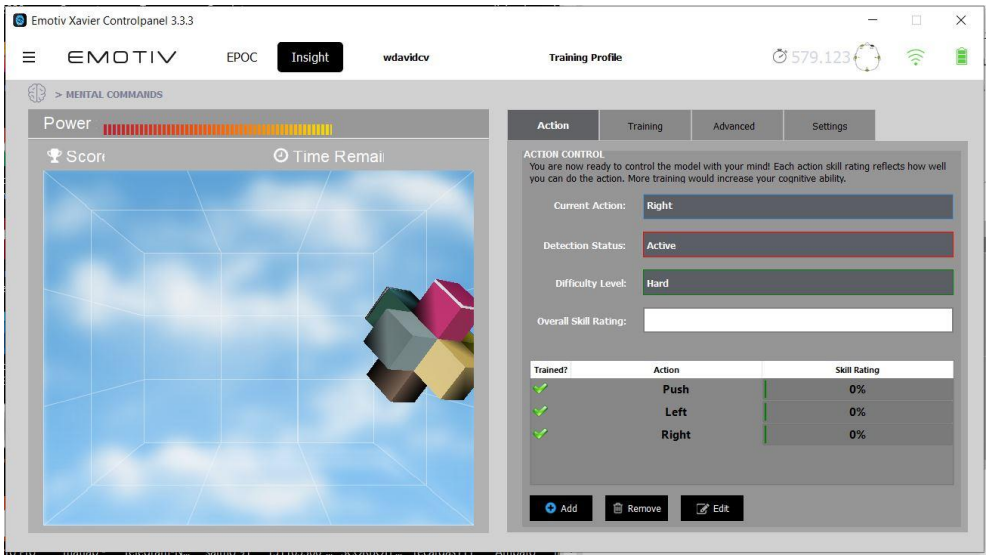


Figura 15: Interfaz de comandos mentales

Elaborado por: El investigador

Interfaz de expresiones Faciales

En la interfaz de expresiones faciales se logra reconocer movimientos como:

- Muecas
- Sonreír
- Parpadeo
- Sorpresa
- Fruncir Ceja
- Expresiones de Enojo

En la figura 16 se puede observar los parámetros de sensibilidad.

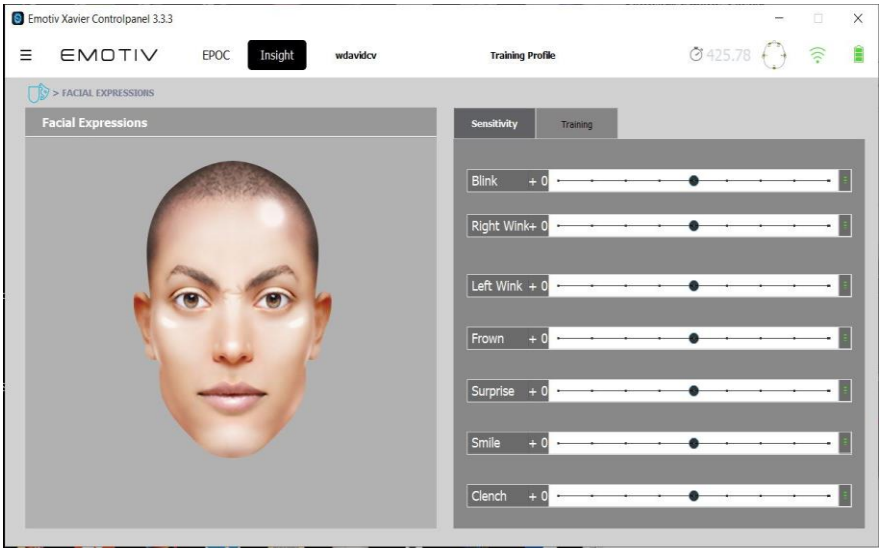


Figura 16: Expresiones faciales

Elaborado por: El investigador

Comunicación

Tecnología inalámbrica

La transmisión de los datos del sistema de manipulación y monitoreo requiere de una comunicación inalámbrica, con el fin de reducir al minino la cantidad de cables utilizada, de manera que los errores por desconexión se reduzcan prácticamente en su totalidad y el proyecto no presente obstáculos o molestias de ningún tipo.

Tabla 10. Tecnología inalámbrica

Parámetros	**Bluetooth**	**Wifi**	**ZigBee**	**Radio Frecuencia**
Descripción				
Banda Operación	2.4 GHz	2.4 GHz	2.4 GHz	2.4 GHz
Alcance de transmisión	10 m	150 m	500 m	90 m
Ancho de banda	1 – 32 Mbps	11 – 300Mbps	20 – 250 Kbps	50 – 70 Mbps
Topología	Picored	Malla, estrella	Malla	Punto a Punto
Consumo de Potencia	40 mA	400 mA	30 mA	< 150 mA

Elaborado por: El investigador

La tabla 10 muestra la comparación entre diferentes tecnologías inalámbricas, según el análisis comparativo se determinó que la tecnología adecuada para el desarrollo del sistema de investigación es la tecnología wifia en la banda operación 2.4 GHz en vista que cumple los requerimientos adecuados para la aplicación, como la tasa transmisión, el alcance, velocidad, consumo de energía, entre otras etc.

Router TP-LINK

El sistema está centrado en una red inalámbrica wifi que conecta el servidor MQTT con la etapa de procesamiento y control de datos, para ello se utiliza un Router TP – LINK, el mismo que se observa en la figura 17; consta de dos antenas, trabaja bajo el protocolo IEEE 802.11 b/g/n a una frecuencia de operación de 2.4 Ghz, 5 Ghz en una banda única, con una velocidad de transmisión de 300 Mbps, ofrece además una conexión estable y seguridad avanzada para la red

Figura 17. Router TP – LINK

Elaborado por: El investigador

Protocolo de Comunicación

El protocolo de comunicación para el presente proyecto es el MQTT puesto que, a más de las ventajas mencionadas en el capítulo 1 en fundamentación teórica de la presente investigación, es de tipo message queue, esto quiere decir que le router genera una cola de mensajería única para cada uno de los clientes, estos mensajes se mantienen como recibidos hasta que son entregados ante fallos de conectividad. En la arquitectura del sistema, el router genera una red de comunicaciones inalámbrica local y conecta todos los elementos a un ordenador de placa reducida (Rasberry Pi 3B+) que actúa como un concentrador del protocolo MQTT, el cliente, en el presente caso, es la placa ESP32, el Arduino mega y el programa Unity

Fuentes alimentación de Brazo robótico Scorbot

Los cinco ejes y la pinza del robot son operados por servomotores de corriente continua. La dirección de la revolución del motor está determinada por la polaridad del voltaje de funcionamiento: con el voltaje de corriente continua positiva gira el motor en una dirección, mientras a un voltaje de corriente continua negativa lo gira en la dirección opuesta. El voltaje de operación no minal de los motores es de 12V DC y los mismos consumen una corriente de 900 mA para desplazar a cualquiera de los ejes sin carga. Se realiza la suma de las corrientes que consumirán desde la fuente.

$$Corriente\ Total = Corriente\ de\ los\ servomotores + Corriente\ de\ Puentes\ H$$

$$Corriente\ Total = 900mA + 300mA$$

$$Corriente\ Total = 1.2\ A$$

Figura 18. Fuente de voltaje

Elaborado por: El investigador

Driver de Motor

El módulo de control de motores es un dispositivo que permite controlar el sentido de giro, así como también la velocidad de 2 motores de corriente directa en dependencia de las señales enviadas del ordenador de bajo costo (BSC). En este proyecto se utilizó el módulo de puente H L298N que permite el control de una variedad de los servomotores de corriente directa. Su componente el L298N se compone de 2 puentes H independientes con capacidad de conducción de 2 amperios constantes.

Este driver nos permite el control de los servomotores que dan la movilidad a las articulaciones del brazo robótico scorbot, con lo cual se puede tener el control ya que se envía la modulación por ancho de pulso (pwm) permitiendo variar este parámetro.

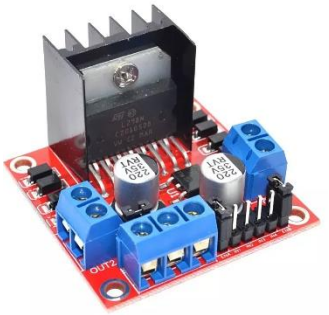


Figura 19. Modulo L298N

Alimentación de los Actuadores

Silla de ruedas

En el caso de los actuadores es recomendable el uso de baterías secas o de ácido-plomo, son las más utilizadas, por su facilidad de recargarse. Los 2 motores de 12 V de corriente continua positiva que se utiliza en la silla de ruedas consumen 5 A cada uno. El cálculo de duración de la batería se lo realizo como en el caso anterior. La suma de las corrientes que consumirá la batería.

$$Corriente\ Total = Corriente\ de\ los\ Motores + Corriente\ de\ los\ Reles\ industriales + Corrientes\ de\ los\ diodos + Corrientes\ de\ los\ transistores$$

$$Corriente\ Total = 10\ A + 300\ mA + 5\ mA + 500mA$$

$$Corriente\ Total = 10.85\ A$$

Figura 20. Batería Acido - Plomo

Elaborado por: El investigador

3.2.4 Implementación del Sistema

El trabajo de investigación aborda las siguientes etapas:

Etapa de Sensorización

El primer paso es conectarse al ordenador mediante la tecnología inalámbrica bluetooth, luego se envía el método de inicio de sesión, para lo cual se declarar una librería liberada por Neuroskyn. En la figura 21 se observa el algoritmo, el cual verifica la conexión del electrodo hacia el puerto de la PC. A demás se configura la conexión en cuanto a velocidad y los tamaños de paquetes, y se añade la librería de extensión "dll".

```
loadlibrary('thinkgear64.dll','thinkgear64.h');
fprintf ('thinkgear64.dll loaded\n');
dllVersion = calllib ('thinkgear64', 'TG_GetDriverVersion');
fprintf ('ThinkGear DLL version: %d\n', dllVersion);
connectionId1 = calllib ('thinkgear64', 'TG_GetNewConnectionId');
if (connectionId1 < 0) error (sprintf ('ERROR: TG_GetNewConnectionId
() returned %d.\n', connectionId1));
end;
errCode = calllib ('thinkgear64', 'TG_Connect', connectionId1,
comPortName1, TG_BAUD_57600, TG_STREAM_PACKETS);
if (errCode < 0)
end
fprintf ('Connected.  Reading Packets...\n');
if (calllib ('thinkgear64','TG_EnableBlinkDetection', connectionId1,1)
==0)
disp('Blinkdetection');
end
```

Figura 21. Conexión del electrodo Mindwave con Matlab

Elaborado por: El investigador

Una vez iniciado la conexión se adquieren los datos, en la figura 22 se observa el código correspondiente a la adquisición de los paquetes de la onda cruda. En la primera condición se verifica la llega del paquete, en la segunda condición se comprueba si es el paquete es el correcto y se adquiere los datos útiles.

```
i=0;
j=0;
disp ('Reading Brainwaves');
t = timer ('TimerFcn', 'stat=false; disp(''FIN'')','StartDelay',40);
errCode = calllib ('thinkgear64', 'TG_EnableAutoRead', connectionId1,-
1);
start(t)
stat=true;
while (stat==true)
if (calllib ('thinkgear64','TG_GetValueStatus', connectionId1,
TG_DATA_RAW) ~= 0)
data_att(j,1) = calllib ('thinkgear64','TG_GetValue', connectionId1,
TG_DATA_ATTENTION);
data_att(j,2) = calllib ('thinkgear64','TG_GetValue', connectionId1,
TG_DATA_MEDITATION);
data_att(j,3) = calllib ('thinkgear64','TG_GetValue', connectionId1,
TG_DATA_RAW);
disp(data_att(j,1));
Plot Graph
plot(data_att(:1),'-r','Linewidth',1);
title('Attention');
disp(data_att(j,2));
Plot Graph
```

Figura 22. Adquisición de datos con Matlab del sensor Mindwave

Elaborado por: El investigador

Neuroskyn ha implementado un algoritmo que puede medir los niveles de relajación y atención en una escala de 0 uV a 100 uV, todo esto a través de ritmos cerebrales. En figura 23, los datos receptados de la diadema reciben diferentes tipos de información en sus paquetes de comunicación, se detallan a continuación:

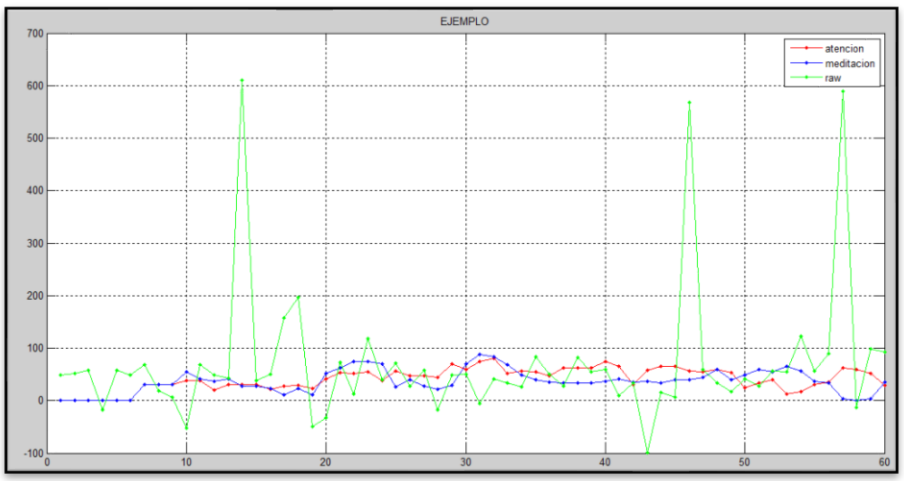


Figura 23: Ondas cerebrales: atención, meditación y raw

Elaborado por: El investigador

.ATENCIÓN eSense (nivel de concentración)

Los datos de concentración se envían en un byte sin signo, el rango de este nivel es de 0 a 100. Cuando el valor es de 1 a 40 indica que el nivel de concentración es muy bajo,

nivel entre 40 y 60 es neutral, cuando esta entre 60 y 80 se considera un poco elevado y valores entre 80 y 100 son considerados elevados, esta señal se transmite cada segundo.

MEDITACIÓN eSence (nivel de calma)

Meditation eSence es un byte sin signo, este dato indica los niveles de calma y relajación, los valores de medida de estas señales son de 0 a 100 y son transmitidas cada segundo.

RAW (Muestra de onda cerebral)

Es un valor que consiste en dos bytes, su valor se encuentra en una variable de 16 bits con signo, su rango va en el intervalo de -32768 a 32767. En el primer byte representa los niveles altos, mientras en el segundo indica los niveles bajos. Esta señal transmitida por la diadema se da 512 veces por segundo.

Estructura de los paquetes

Una vez establecida la conexión entre la diadema y el ordenador se empieza a recibir los paquetes que la diadema proporciona. Estos paquetes son transmitidos en forma asíncrona en serie y en cadena de byte. Cada paquete está desarrollado por tres partes como se indica en la figura24.

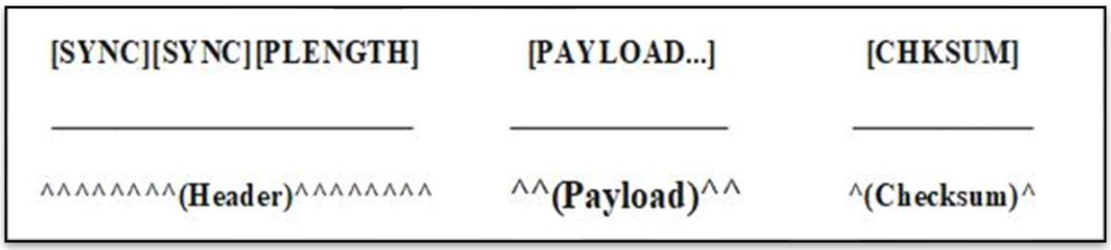


Figura 24. Estructura de los paquetes Mindwave

Cabecera (header)

La cabecera está formada por dos bytes “SYNC” y un byte “PLENGTH”, los dos primeros bytes de sincronización deben ser “0xAA”, se envía dos bytes seguidos para evitar fallos en la recepción. El “PLENGTH” indica la longitud en bytes. Este valor puede ser entre 0 y 169 cualquier valor superior indicara que surgió un error.

Carga útil (Payload)

La carga útil es un conjunto de bytes, en donde se realiza la decodificación de los paquetes, para dicha decodificación primero tiene que haberse dado por valido el paquete tras la comprobación del checksum.

Checksum

El checksum es un byte que se utiliza para la integridad de los paquetes, es la suma de todos los bytes del paquete dentro de una carga útil, se toman 8 bits más bajo y se le realiza el complemento A1. Es decir, se debe realizar esta operación con cada paquete y verificar que los valores coincidan con el bit del "CHECKSUM", si los valores no coinciden el paquete se descarta, si el paquete coincide se le toma como bueno. Para que el paquete de recepción de datos sea válido se debe realizar estos tres pasos para comprobar la suma de la carga útil de los datos recibidos.

Etapa de Procesamiento

Neuroskyn

Después de la adquisición de las señales, se extrae un patrón único para cada uno de los movimientos de las articulaciones. Para detectar un evento del parpadeo de un ojo de las señales del sensor blink, se utilizó una función matemática filtra picos máximos que exceden un valor de umbral de 600 uV, este umbral se determinó experimentalmente. Así, cada vez que las señales superan los umbrales, la función envía un mensaje al panel de control desplazándose aleatoriamente las opciones de movimiento como se observa en la figura 25.

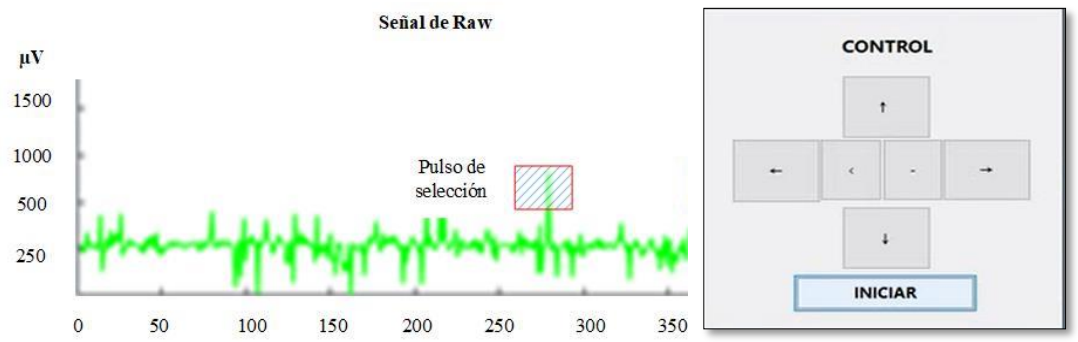


Figura 25. Valor umbral de parpadeo

Elaborado por: El investigador

Para determinar el movimiento hacia a la izquierda los picos máximos de la señal de atención deben superar un valor umbral mayor a 80 uV. En la figura 26 se observa el valor umbral.

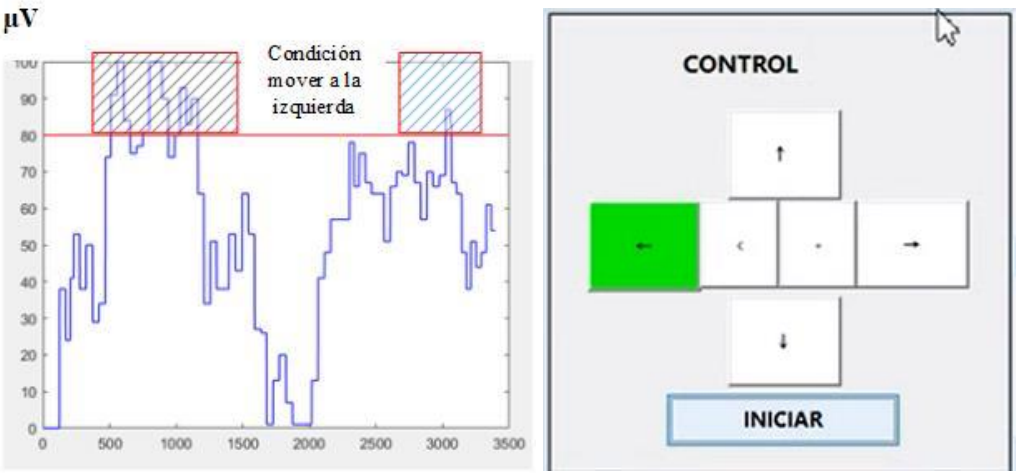


Figura 26. Condición mover a la izquierda de la señal de atención

Elaborado por: El investigador

De manera similar, para determinar el movimiento hacia la derecha, el máximo de los picos debe estar en rango de 60 uV y 80 uV de la señal de atención. En la figura 27 se observa el valor umbral del movimiento a la derecha.

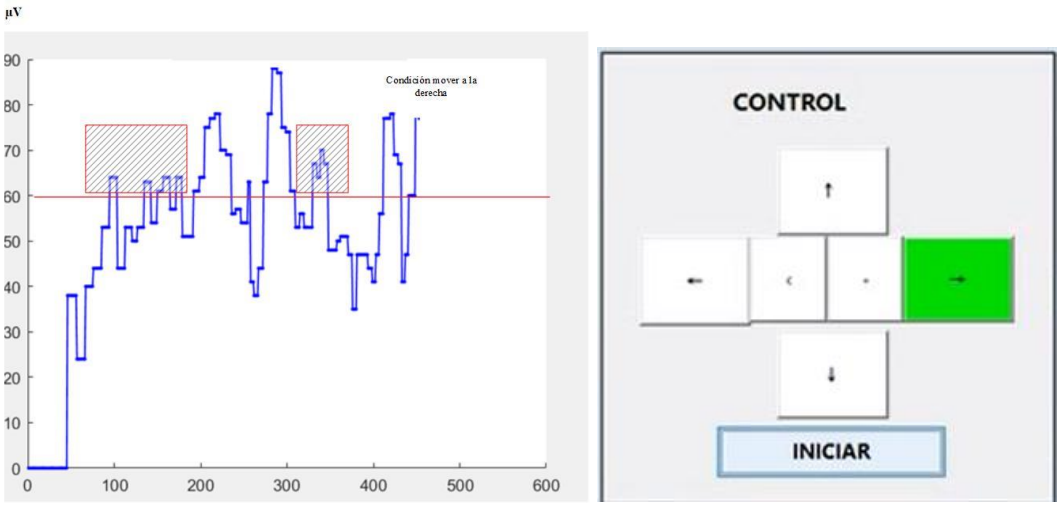


Figura 27. Condición mover a la derecha de la señal de atención

Elaborado por: El investigador

La siguiente señal de control utilizada proviene de la señal de meditación, la relajación de la persona produce cambios en la amplitud de la señal de meditación. La condición

de subir o ascender cuando se genera un pico mayor 80 uV. Como se puede observar la figura 28.

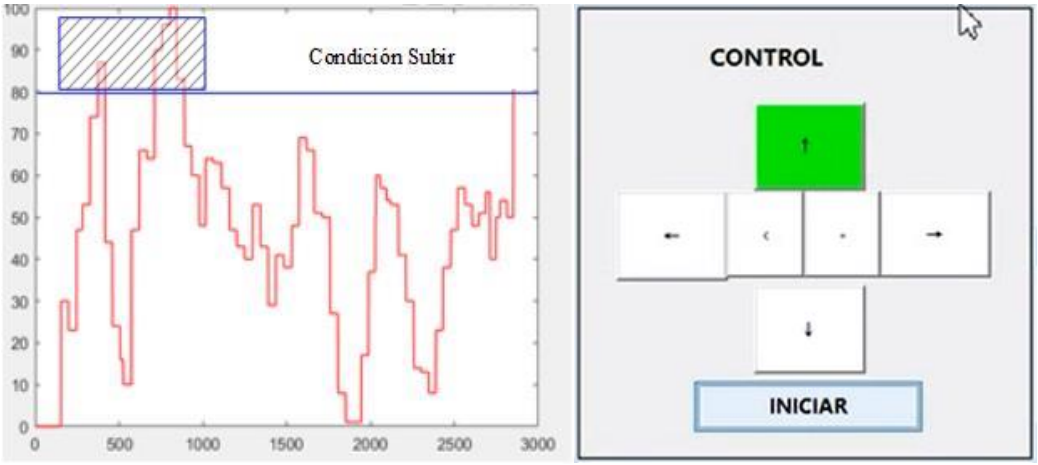


Figura 28. Condición subir de la señal de meditación.

Elaborado por: El investigador

El movimiento hacia abajo, se detecta utilizando los picos de la señal de meditación en el rango de 60 uV a 80 uV. En la figura 29 se observa el umbral seleccionado

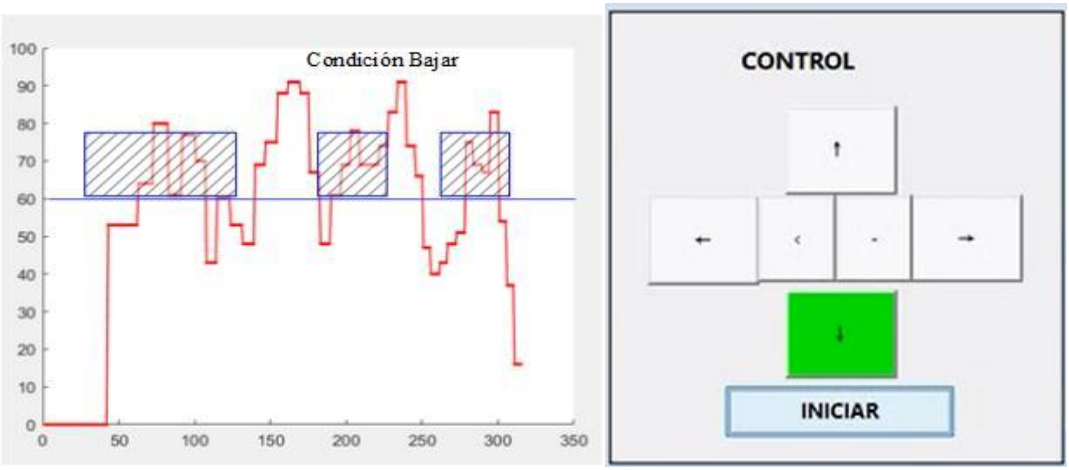


Figura 29. Condición Bajar de la señal de meditación.

Elaborado por: El investigador

Para la apertura de la abrazadera del manipular, se utiliza la señal de atención comprendida en el rango de 40 uV y 60 uV, como se observa en la figura 30.

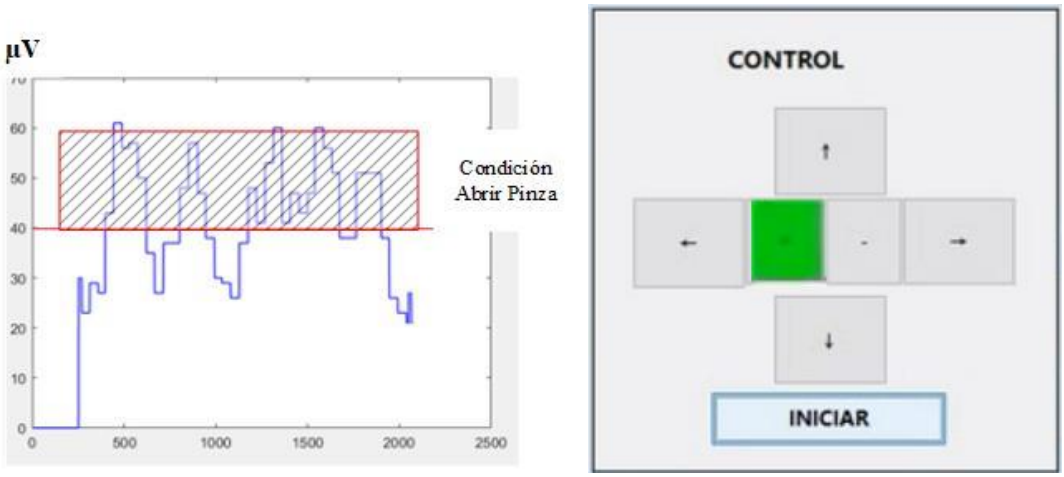


Figura 30. Condición abrir pinza o apertura.

Elaborado por: El investigador

Por último, tenemos la condición de cerrar pinza está comprendido en rango de 40 uV a 60 uV de la señal umbral de meditación, en la figura 31 se muestra las condiciones.

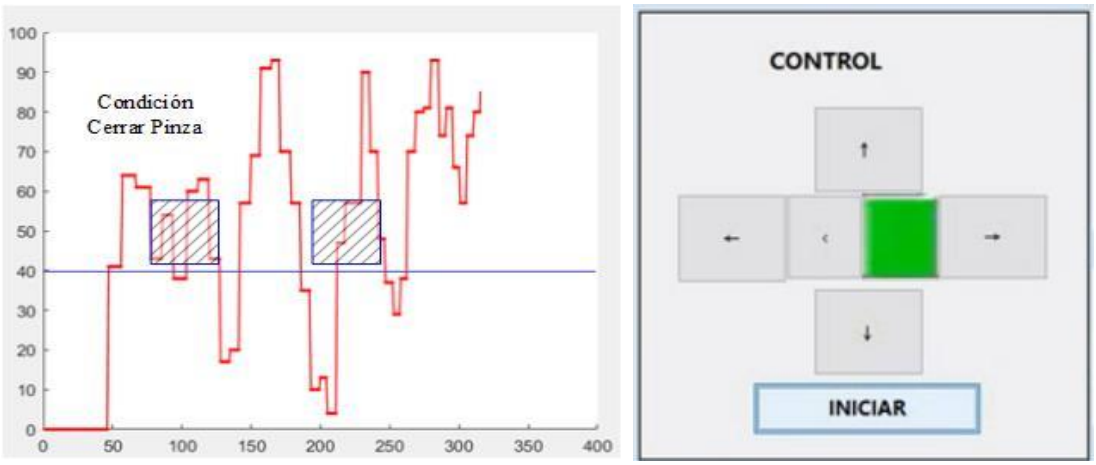


Figura 31. Condición cerrar pinza.

Elaborado por: El investigador

Emotiv Emokey

Es un software conectado al panel de control emotiv. Se traduce en entradas de teclado para el uso de programas o controladores externos, el conjunto de traducciones definidas se llama “Mapeo de Emokey” y se puede guardar para su reutilización.

Este software se divide en dos partes: reglas y condiciones. Las reglas definen que caracter se enviara a la cola de entrada del sistema operativo. Cada regla puede tener múltiples condiciones, estas condiciones definen lo que debe suceder, para que la regla pueda activarse, en la figura 32 se observa el panel emokey.

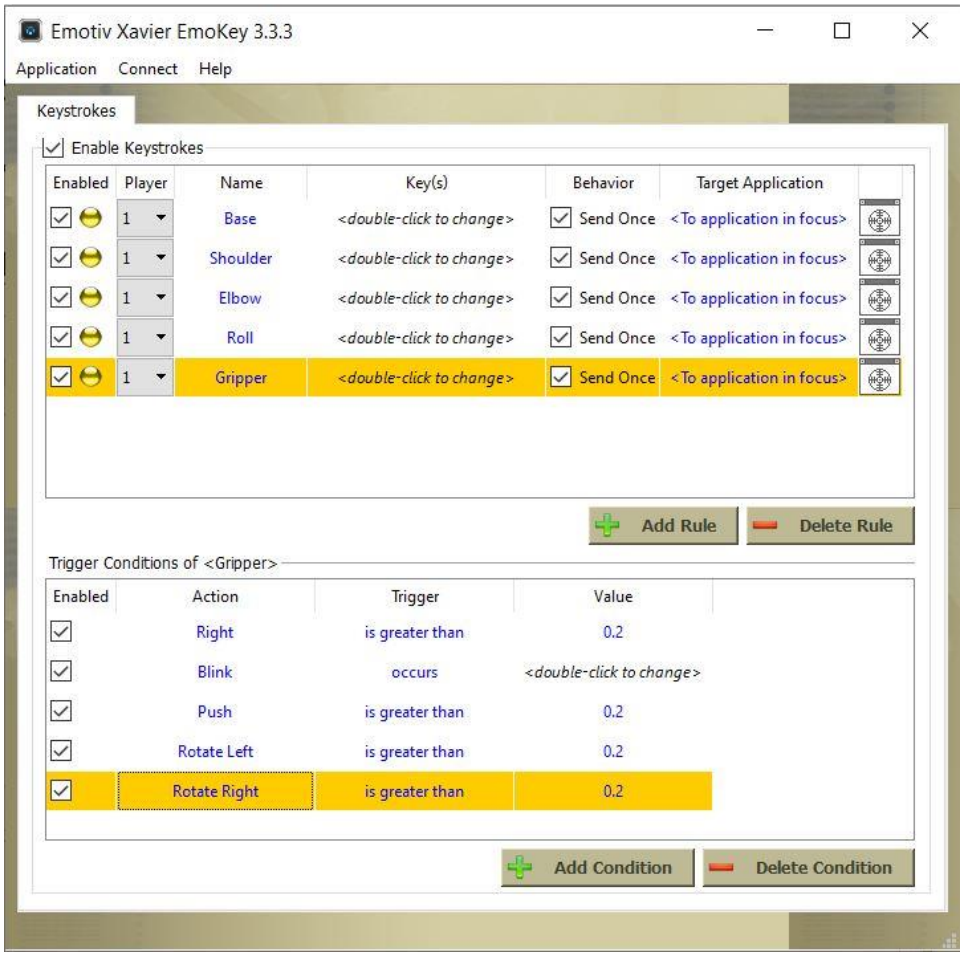


Figura 32. Panel Emotiv Emokey

Elaborado por: El investigador

Estas condiciones reconocidas por el panel de control emotiv, se envían mediante el Emokey a la interfaz de control del Matlab: hacia las condiciones mover izquierdo,

mover derecho, subir, bajar, abrir pinza y cerrar pinza. Los caracteres que envía el software emokey se visualizan en la tabla 11.

Tabla 11. Caracteres enviados por el software Emokey

Software Emotiv		**Software Matlab**	
Panel de Control Emotiv (Acción)	**Emotiv Emokey (Caracteres)**	**Iconos**	**Condición**
left	a	←	Mover izquierdo
right	b	→	Mover derecho
Rotate left	c	↑	subir
Rotate rigth	d	↓	bajar
pull	e	‹	Condición abrir
Blink	f	·	Condición de cerrar

Para la configuración de envío de los caracteres del emokey hacia el Matlab, se hace uso de la librería emotiv sdk (edk.dll), debido a que contiene las API (interfaz de programación de aplicaciones) permitiendo la comunicación entre dos aplicaciones, se realiza presionando doble clic sobre la celda Target Application (Aplicación Destino), esto visualizara el dialogo de la figura 33.

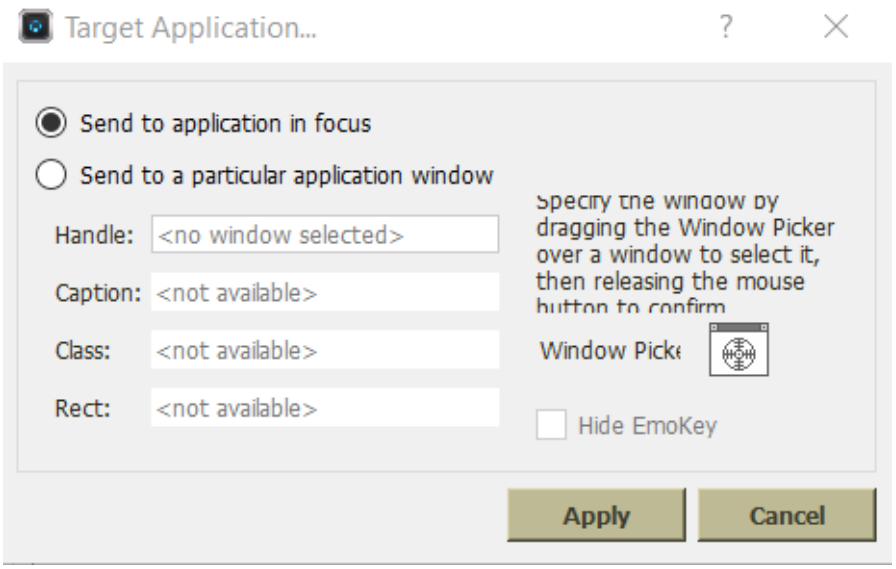


Figura 33. EmoKey - Selección del programa destino

Elaborado por: El investigador

En donde se procede a seleccionar como destino la aplicación que tiene el foco, o bien, una en particular arrastrando y soltando el icono del target Application sobre el recuadro de la interfaz gráfica, como se observa la figura 34.

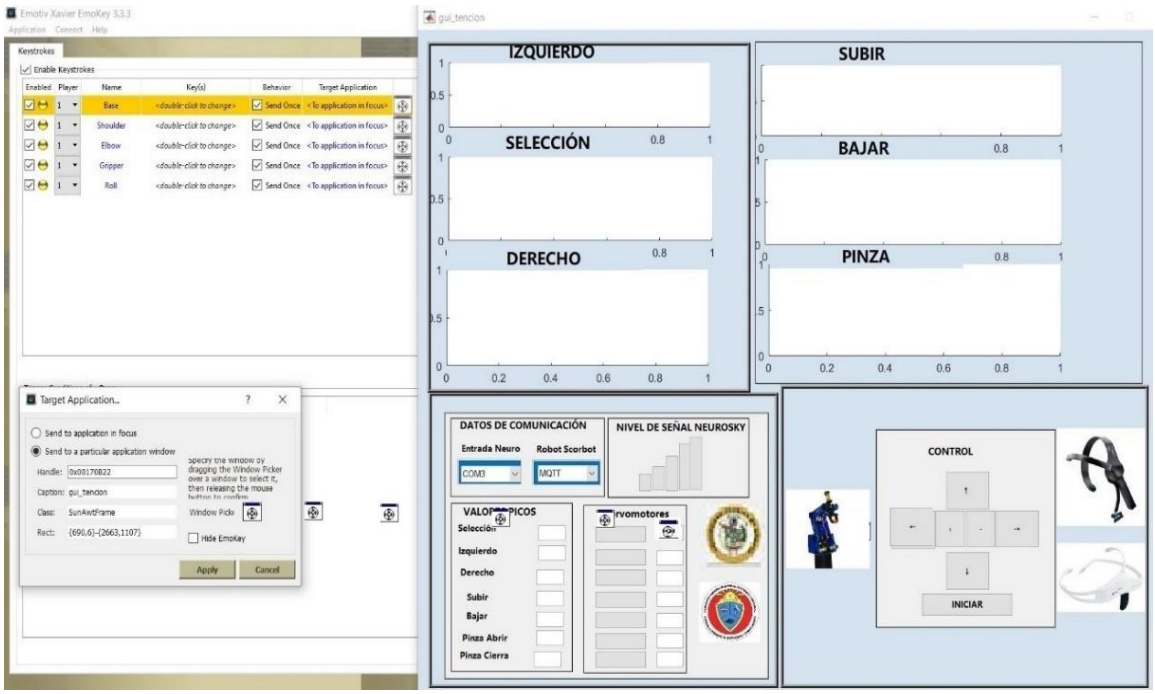


Figura 34. Configuración de envió de datos al Matlab

Elaborado por: El investigador

Para que la interfaz de Matlab reciba los caracteres de Emotiv Emokey, utiliza el código descrito en el anexo 4.

Entrenamiento

Después de desarrollar el procesamiento de las señales, se procede a realizar una variedad de pruebas repetitivas mientras piensa en una acción. Las acciones varían de acuerdo a los movimientos por ejemplo movimiento hacia la izquierda o hacia a la derecha, o hacia arriba y etc., y cuando se realiza cierta acción se visualiza el número de aciertos en el panel de valores pico como se observa en la figura 35.

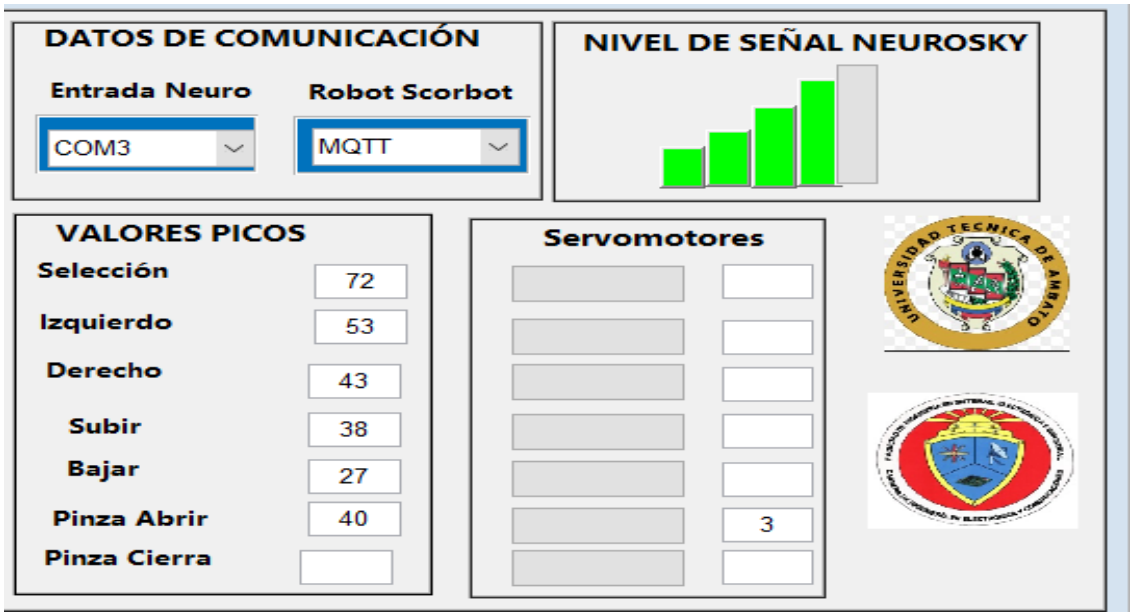


Figura 35. Panel Valores

Elaborado por: El investigador

Se probo el sistema durante 20 pruebas en tiempo real y se obtuvo una precisión general del 85.83 % como se muestra en la tabla 12. El comando con mayor precisión es subir o ascender y los comandos con menor precisión es abrir y cerra.

Tabla 12. Precisión de todos comandos

Comandos	**Número de intentos correctos de 20**	**Porcentaje**
Izquierdo	15	85 %
Derecho	15	85 %
Subir	16	95 %
Bajar	14	80 %
Cerrar	13	90%
Abrir	13	80 %
Precisión general		85.83 %

Elaborado por: El investigador

Visualización

En este apartado se considera los datos del dispositivo en este caso la diadema neurosky, de tal manera que dicha información es procesada para visualizar las señales. Para el presente proyecto de investigación, la interfaz de monitoreo se realizó en guide de Matlab, la cual recepta los datos provenientes del sensor de la parte frontal del cerebro, y se visualiza en la venta como se indica en la figura 36.

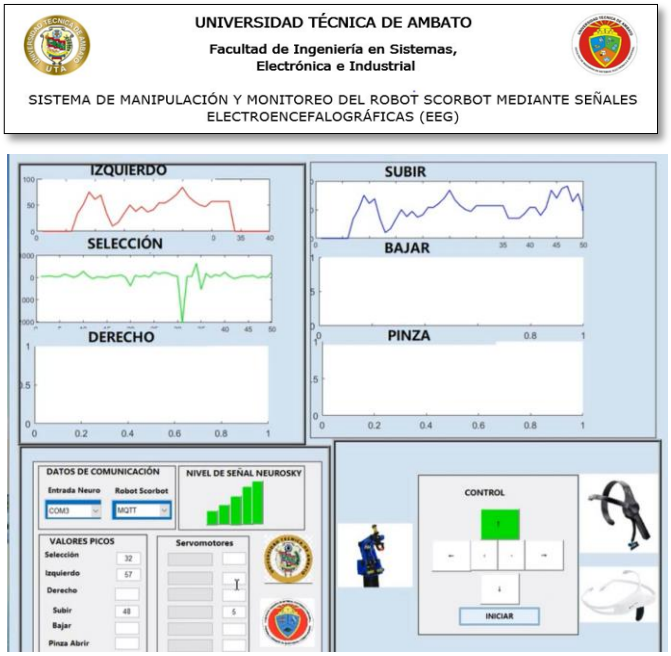


Figura 36. Panel Principal

Elaborado por: El investigador

En este panel recepta la información de:

- Datos de Comunicación
- Señales de la actividad de movimiento
- Señales de parpadeo

Datos de Comunicación

En este panel la comunicación entre MATLAB y los cascos de señales EEG establecen una comunicación por el puerto COM. La transferencia de datos no se realiza a menos que seleccione el puerto COM3. Y la comunicación interna entre la Raspberry se realiza mediante le protocolo de comunicaciones MQTT, además se muestra en esta sección los niveles de señal del casco, también los niveles de señales que se producen

al realizar la selección del movimiento. En el anexo 5 detalla el método para comunicación MQTT.

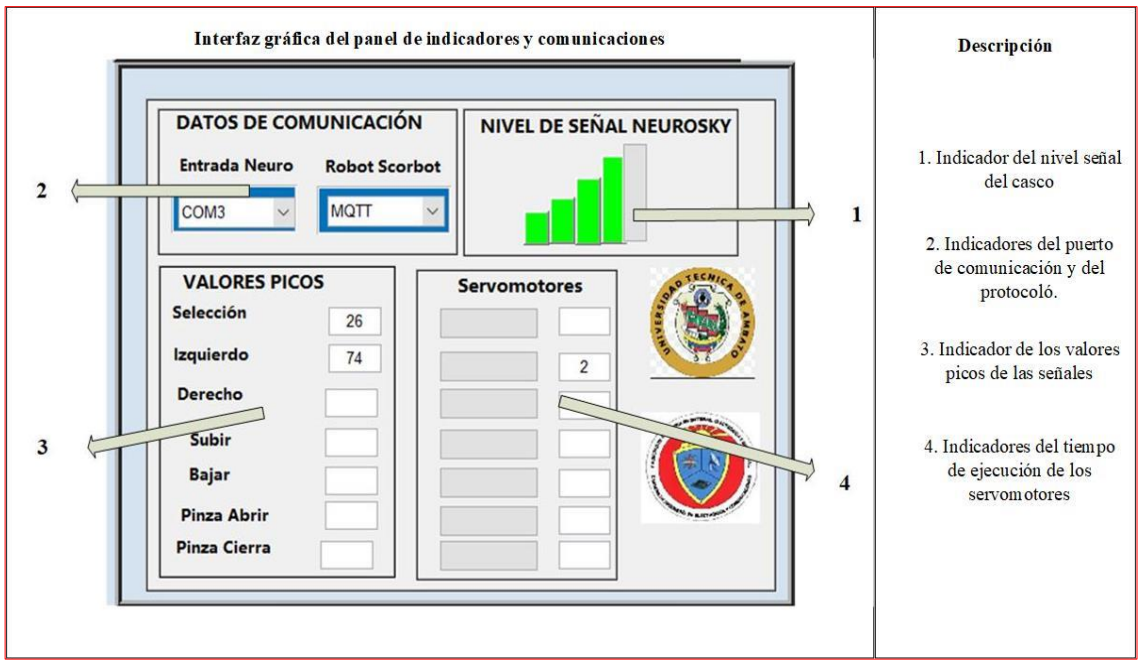


Figura 37: Panel de Comunicaciones

Elaborado por: El investigador

Inicialización de la Diadema NeuroSky

La inicialización de la diadema NeuroSky está formado por la librería:

ThinkGear64

La libreria ThinkGear64 es necesario para que las aplicaciones puedan conectarse a los auriculares. El TGC es obligatorio porque algunas plataformas de software no permiten el acceso directo a los puertos COM de la computadora. TGC se conecta específicamente al puerto COM de los auriculares permitiendo la conectividad de las aplicaciones.

Inicialización del publicador MQTT

El método de envió de datos hacia el bróker desarrollado en Matlab se inicializa y se ejecuta cuando las señales superan satisfactoriamente las condiciones para el movimiento. Dentro de la configuración se identifica la respectiva librería (javaaddpath('jars/iMqttClient.jar'), el topico (PUB_TOPIC_1 = 'casa'), la ip del broket

(mqttinterface = MqttInterface('matlab_mqtt_node', '172.88.1.104', 1883, 10) y el mensaje (pub_msg_1). En anexo 6 se detalla el algoritmo correspondiente a su configuración.

Señales de parpadeo

La señal del parpadeo o señal de selección de movimiento consta de un contador que cuenta los pulsos en alto que superen los 600 uV como se indica en la figura 38, además de presentar la señal oscilante.

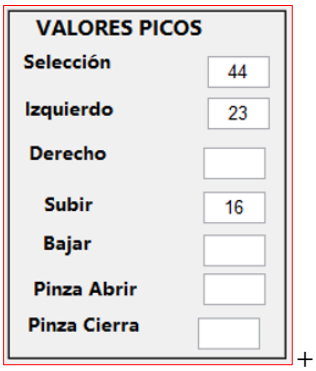


Figura 38: Indicador de pulsos en alto

Elaborado por: El investigador

Señales de la actividad de movimiento

En este panel se visualiza las señales que se produce al activar el icono de movimiento, mostrando los picos que se genera al enfocarse el pensamiento en mover izquierdo, derecho, subir, bajar, abrir o cerrar pinza. También nos proporciona los datos al activar los respectivos actuadores en nuestro caso los servomotores. Como se observa en la figura 39.

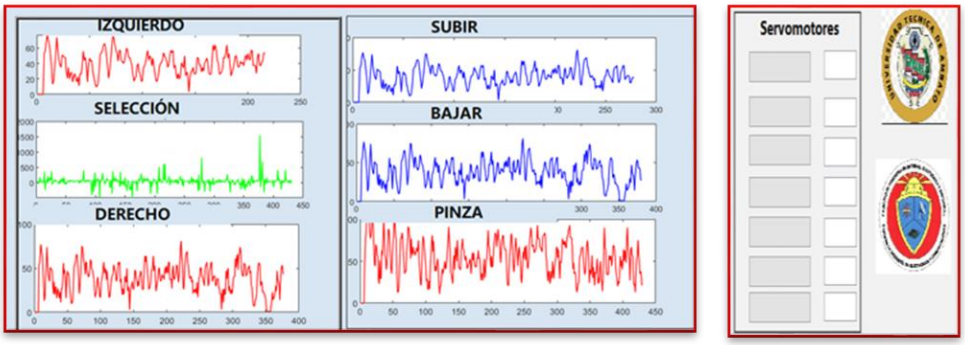


Figura 39: Panel de visualización de señales

Elaborado por: El investigador

Interfaz de Comunicaciones

La interfaz de comunicaciones constituye el protocolo y el medio de transmisión utilizados para la transferencia de datos entre el ordenador y la tarjeta controladora. El sistema de comunicaciones se realiza utilizando la tecnología Wifi como medio de transmisión para la interfaz de comunicación.

Instalación del protocolo MQTT en la raspberry PI

La Raspberry Pi es quien recibe la información mediante su dirección "172.88.1.104" actuando como bróker, se utilizó el protocolo de comunicación Mosquitto, para la instalación y configuración de MQTT en la Raspberry Pi, primero actualizamos el sistema de la Raspberry Pi. En la tabla 13 se muestra el código correspondiente a la actualización de paquetes e instalación del Broker Mosquito.

Tabla 13: Líneas de Código para la instalación Mosquitto

Código	Descripción
sudo apt-get update	La línea de código ejecuta la actualización de los repositorios y paquetes.
sudo apt-get install –y mosquito mosquito-clients	Instala el Mosquitto Broker en Raspbian (Raspberry Pi)
sudo systemctl enable mosquitto.service	Se ejecuta el bróker Mosquito.

Elaborado por: El investigador

Se utilizó la Raspberry Pi como bróker, para obtener la información que los publicadores envían, este administra y reenvía la información a los subscriptores.

3.2.5 Etapa de Control Electrónico

Controlador Rasberry PI

Una vez finalizado el entrenamiento en panel de control se realiza la conexión con los scripts de Python, que son los responsables de leer los comandos y enviarlos al robot, su funcionamiento se basa en la activación de los pines GPIO utilizados para enviar y recibir señales digitales, para controlar cada GPIO, se utilizó la codificación de clases en el lenguaje Python.

Control GPIOS

La Figura 40 mediante diagramas UML, muestra las clases utilizadas para controlar los pines GPIO de la Raspberry Pi, cada clase tiene sus propiedades y métodos, es importante enfatizar que existe una relación de composición entre la clase “control” y el resto de clases.

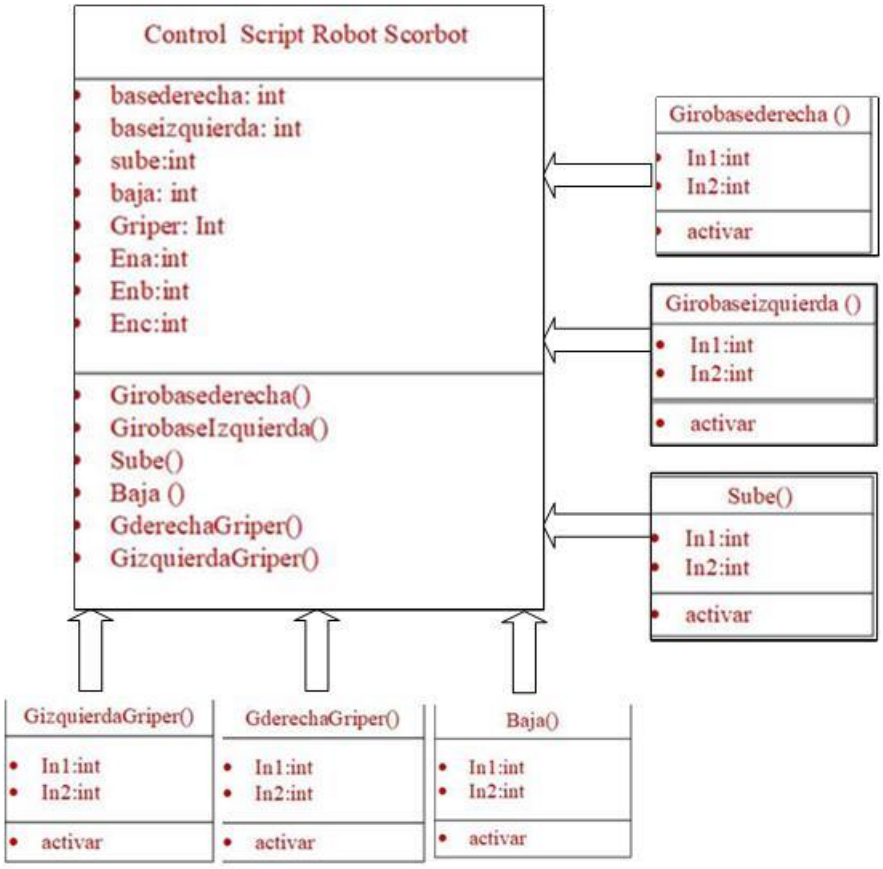


Figura 40: Diagrama control Rasberry PI

Elaborado por: El investigador

En la tabla 14 se ilustra las características del diagrama de control de la Rasberry PI de cada clase y subclase.

Tabla 14: Descripción de las Clases Robot Scorbot

Clase	Descripción
Control script Robot scorbot	Proporciona a las subclases los atributos para el control de los pines GPIO.
Girobasederecha ()	Acciona al motor de la base del manipulador Scorbot en sentido derecho.
Girobaseizquierda ()	Acciona al motor de la base del manipulador Scorbot en sentido izquierdo.
Sube()	Acciona al motor del codo del brazo Scorbot en un sentido ascendente.
Baja()	Acciona al motor del codo del brazo Scorbot en un sentido descendente.
GderechaGriper()	Acciona al motor del gripper del manipulador Scorbot en un sentido derecho.
GizquierdaGriper()	Acciona al motor del gripper del manipulador Scorbot en un sentido izquierdo.

Conexión MQTT

El diagrama de comunicación mqtt está representado en la Raspberry Pi, tiene atributos de tipo mqtt, como cliente y datos de usuario, así como atributos de tipo String. Estos atributos definen la dirección IP del Broker, el nombre del tópico y el mensaje recibido. El método utilizado es crear una conexión (on_conecct), recibir un mensaje (on_message) e inicializar el cliente (mqtt.client). Estos son los parámetros necesarios para establecer comunicación con el entorno Unity bajo el protocolo MQTT a demás establecer la comunicación en el Arduino [26].

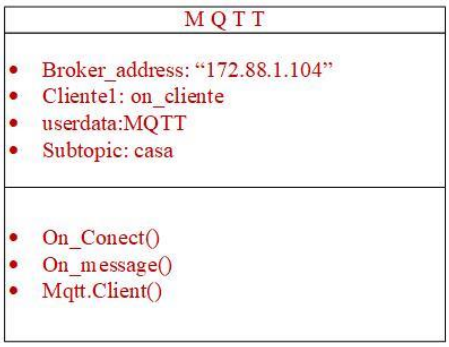


Figura 41: Diagrama de Conexión MQTT

Elaborado por: El investigador

Conexión de los servomotores a los integrados l298n

El brazo robótico está conectado al controlador mediante un cable con un terminal D50, un conector de 50 pines al que se encuentran enlazados todos los cables de energía, motores y finales de carrera. Los cables de los motores y los finales de carrear del brazo se conectan al D50, los cables son flexibles y resisten a las roturas. En la tabla 15, se especifican las conexiones existentes entre los distintos componentes del brazo.

Tabla 15: Descripción de las funciones de los pines del conector D50

11	**Función**	# **Pin**	**Función**	# **Pin**	**Función**	# **Pin**	**Función**	# **Pin**	**Función**
1	Pulso B del encoder	**2**	Pulso B del encoder	**3**	Pulso A del encoder del eje 5	**4**	Pulso A del encoder del eje 3	**5**	Pulso A del encoder del eje 1
6	Final de carrera del eje	**7**	Final de carrera del eje	**8**	Final de carrera del eje	**9**	Voltaje del encoder del eje	**10**	Voljate del encoder
11	Voltaje del coder del eje	**12**	Motor del gripper -	**13**	Motor del eje 5 -	**14**	Motor del eje 4 -	**15**	Motor del eje 3 -
16	Motor del eje 2 -	**17**	Motor del eje 1 -	**18**	Pulso B del encoder del eje	**19**	Pulso A del encoder del gripper	**20**	Pulso A del encoder del eje 4
21	Pulso A del encoder del eje 2	**22**	Final de carrera del eje	**23**	Final de carrera del eje	**24**	Final de carrera del eje	**25**	voltaje del econder eje
26	voltaje del econder eje	**27**	voltaje del econder eje	**28**	GND	**29**	GND	**30**	GND
31	GND	**32**	GND	**33**	GND	**34**	Pulso B del encoder del eje	**35**	Pulso B del encoder del eje
36	Pulso B del encoder del eje	**37**	5Vdc	**38**	5Vdc	**39**	5Vdc	**40**	5Vdc
41	5Vdc	**42**	5Vdc	**43**	5Vdc	**44**	5Vdc	**45**	Motor del gripper +
46	Motor del eje 5 +	**47**	Motor del eje 4 +	**48**	Motor del eje 3 +	**49**	Motor del eje 2 +	**50**	Motor del eje 1 +

Elaborado por: El investigador

En la figura 42 se observa el esquema de conexión electrónica de la interfaz de los puentes H, los integrados L298N son alimentados con una fuente de 12V DC y cada

uno de ellos maneja el giro de un motor, la velocidad angular de los motores se controla con una señal PWM única y las señales de control se conectan de forma directa a la Raspberry PI.

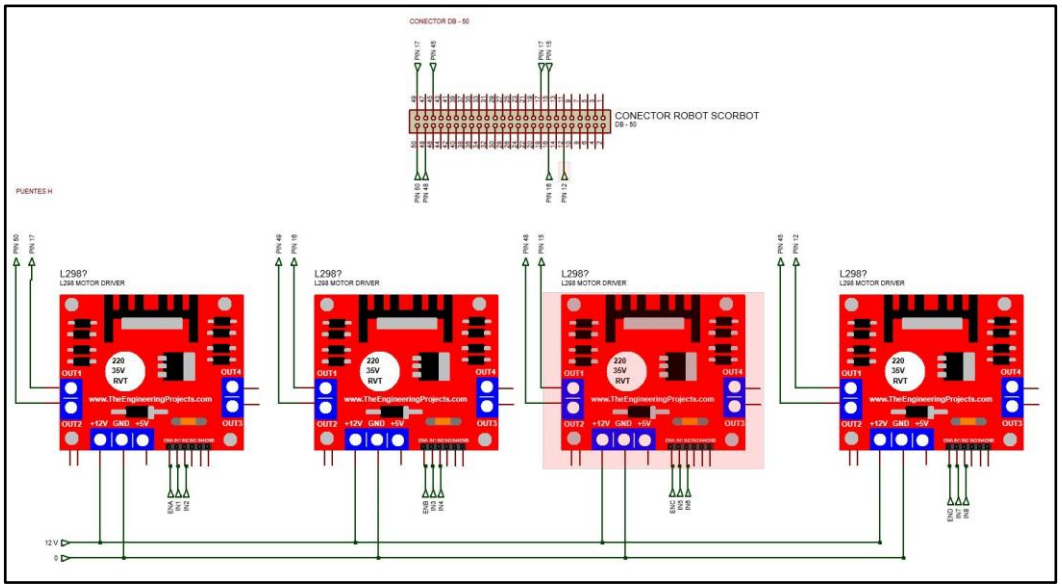


Figura 42: Circuito electrónico de puentes H

Elaborado por: El investigador

Control de servomotores del Brazo robótico scorbot

Los servomotores del brazo robótico, se muestran en la figura 43 estos motores se controlan a través de flujos de mensajes utilizando el protocolo MQTT para ello, se utilizó un ordenador de bajo costo raspberry Pi 3 B+, que utiliza el sistema operativo Raspbian Linux, el sistema de control está basado en Python. Cada servo está controlado por una señal de control diferente, que llega después de suscribirse a la secuencia correspondiente.

El control de los servos se divide en dos partes: el primer control lo dan las corrientes de las señales correspondientes a los sensores de los cascos electroencefalográficas EEG, dentro de los cuales se implementó un algoritmo en el software matlab para cambiar el estado, es decir cada vez que la señal llega debido a la acción de parpadeo del ojo, los servos cambian entre las posiciones establecidas.

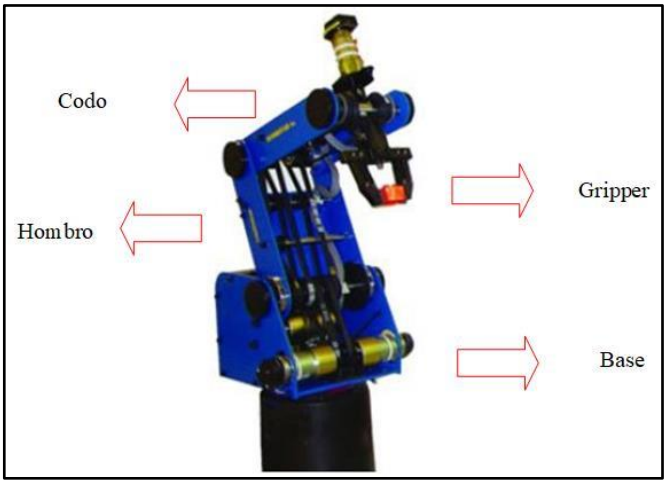


Figura 43: Brazo Robótico y sus articulaciones

Elaborado por: El investigador

El segundo control lo dan las corrientes procedentes del bróker, de la misma forma, se implementó un algoritmo en Python de detección de cambio de estado que permite cambiar la posición de los servos que controlan el movimiento de las articulaciones. Sin embargo, en este caso llegan diferentes señales de control para cada servo. La tabla 16 muestra los canales utilizados, la suscripción al servicio, los mensajes recibidos y la acción del servomotor correspondiente, en el anexo 7

Tabla 16: Control del brazo robótico utilizando el protocolo MQTT

Control del Robot con el casco Neuroskyn			
Canal	**Suscripción**	**Mensaje**	**Acción**
TGAM área FP1	Parpadeo		
Señal de Raw	Condición mover a la Izquierda	a	Base
Señal de atención	Condición mover a la derecha	b	Base
Señal de meditación	Condición subir	c	Shoulder, Elbow
	Condición Bajar	d	Shoulder, Elbow
	Condición abrir	e	Gripper
	Condición Cerrar	f	Gripper
Control del Robot con el casco Emotiv			
Canal	**Suscripción**	**Mensaje**	**Acción**
Giro Y, AccZ	Left	a	Base
Giro X, AccY	right	b	Base
Giro Z, AccY	rotate left	c	Shoulder, Elbow
Giro Z, AccX	rotate rigth	d	Shoulder, Elbow
AF3	pull	e	Gripper
AF4	Blink	f	Gripper

Elaborado por: El investigador

Como se puede apreciar en la figura 44, se observa la interfaz gráfica y el brazo robótico.

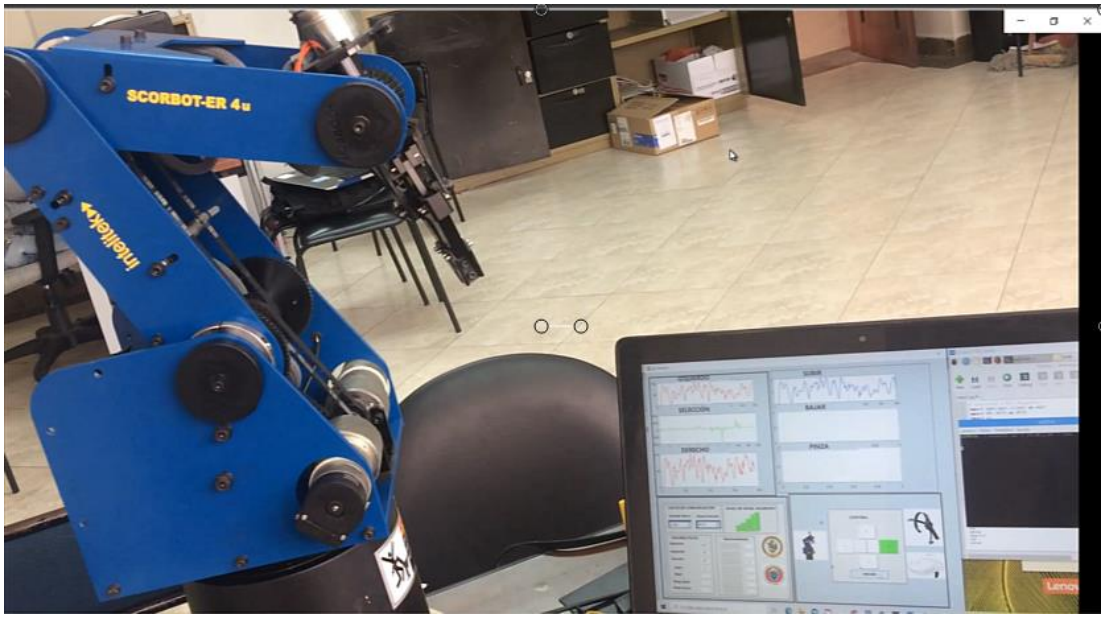


Figura 44: Brazo Robótico y la interfaz grafica

Elaborado por: El investigador

En el proyecto se utilizó las diademas neuroskyn y la diadema emotiv para evaluar el funcionamiento del sistema de control y monitoreo del robot scorbot. Para esta finalidad se tomó tiempos de activación en la interfaz control, evaluando tiempos de respuesta. Dando como resultado tiempos homogéneos en la activación de los movimientos, el tiempo de respuesta estado en función del entramiento previo de cada uno de las diademas. El entrenamiento consiste en enfocar el pensamiento en un determinado objetivo por varias ocasiones, los errores frecuentes que ocurrieron fueron el desgaste de las baterías en Neuroskyn, interrumpiendo la conexión y en el emotiv la sequedad de los electrodos.

El desarrollo del proyecto cumplió con los objetivos planteados, obteniendo un sistema de control y monitoreo del robot scorbot mediante las señales electroencefalográficas, mediante la comunicación TCP aplicando el protocolo MQTT, se puede verificar la eficacia del sistema propuesto, en él envió de datos a los diferentes dispositivos. El proyecto está dedicado a ayudar a las personas con discapacidad total o parcial e integrarse en la sociedad. Este proyecto puede convertirse en un nuevo proyecto de investigación, la interfaz desarrollada para señales EEG se utilizó para manipular los

brazos robóticos Scorbot, prototipos simulados, sillas de ruedas y un vehículo de control remoto.

Control de la silla de ruedas

Es importante tomar en cuenta que las personas discapacitadas empiezan a tener gran dependencia de una tercera para poder movilizarse y cubrir necesidades. Es por ello que se utilizó el sistema de control y monitoreo que se desarrolló en este proyecto de investigación.

Interfaz electrónico de control

Los datos que proporciona el sistema son receptados por la tarjeta electrónica esp32, mediante el protocolo de comunicaciones MQTT, por medio de un algoritmo controla el movimiento de la silla de ruedas. Para ello se diseñó una etapa de acondicionamiento de la señal.

El esquema de la conexión electrónica de la interfaz de los relés se indica en la figura 45, los niveles de 1 lógico de los ESP32 van desde los 2V DC hasta 3.3V DC, en consecuencia, se debe aumentar el voltaje a 5V DC con una corriente 500mA para energizar el relé y dar paso a los 12V DC, se utilizó transistores 2N3904 por su capacidad de amplificación de corriente que va desde 100mA hasta los 500mA, también un diodo 1N4004 conectado en paralelo con la bobina del relé como protección del transistor, se recepta del controlador dos niveles lógicos altos por ende se realizó dos esquemas para cada nivel lógico.

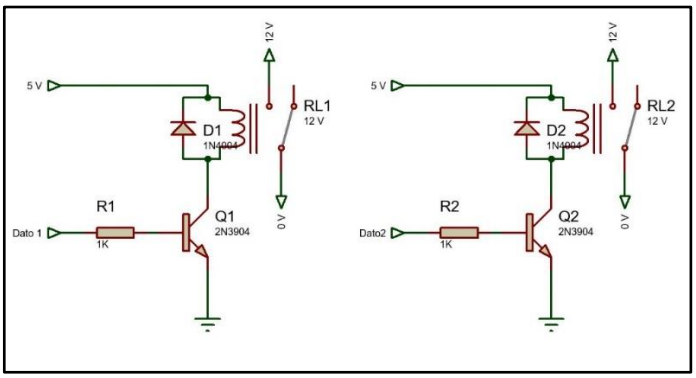


Figura 45: Circuito electrónico de interfaz de relés

Elaborado por: El investigador

La silla de ruedas tiene la capacidad de soportar de 60 Kg a 80 Kg, por consecuencia los motores utilizados para su desplazamiento consumen 12V a 5A cada uno, por esa razón se utilizó contactores industriales. En la figura 46 se ilustra la imagen del sistema de control donde se recepta la señal de 12V del circuito de la interfaz de relés, energizando un relé permitiendo el paso del voltaje de la batería hacia el motor y abriendo el contactor normalmente cerrado del segundo relé cortando el paso de la energía, de la misma manera sucede con el segundo relé recepta el segundo dato permitiendo el paso de la energía y abriendo el contacto del primer relé, de esta manera se controla el cambio de giro de los dos motores obteniendo el desplazamiento hacia delante y hacia atrás.

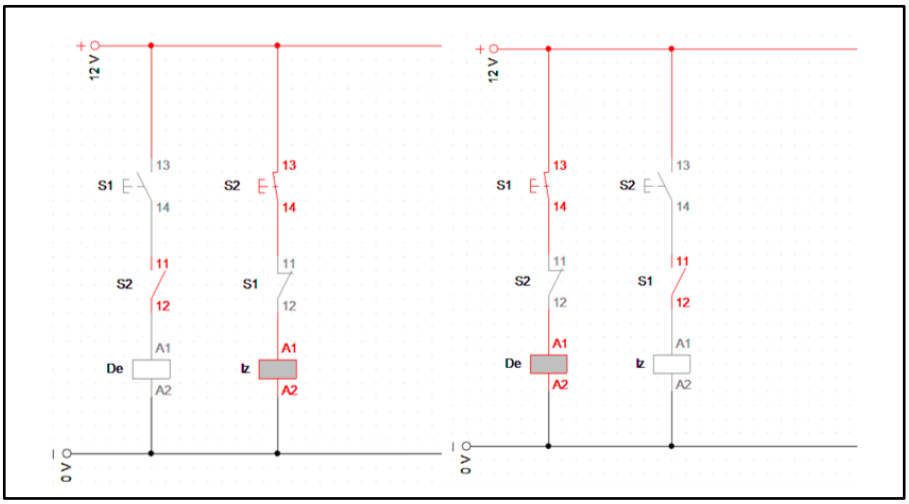


Figura 46: Esquema de control

Elaborado por: El investigador

Movimiento de la silla de ruedas

En la Figura 47 se ilustra la imagen de la silla de ruedas con su respectiva señal, accionando el movimiento de los motores y desplazándose hacia delante, el controlador ESP32 configurado el protocolo MQTT como suscriptor se conectó a la red mediante la tecnología wifi, receptando los pulsos de las señales del casco neuroskyn. Dichas señales pasaron por el circuito de relés, donde fueron amplificadas con la finalidad energizar los relés dando paso a los 12V que posteriormente son utilizadas para accionar los contactores industriales y permitir el movimiento de los motores, en el anexo 8 se puede apreciar el código.

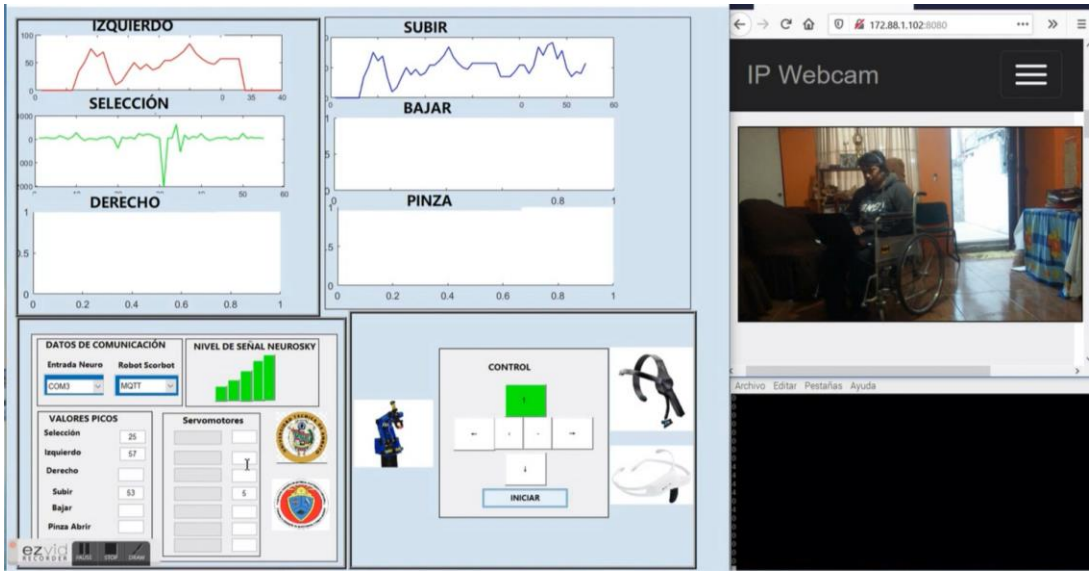


Figura 47: Silla de ruedas controlado por el sistema

Elaborado por: El investigador

Control del brazo robótico Eezy con el Arduino mega

El funcionamiento del prototipo de brazo robótico con el servomotor incluye el uso de una biblioteca de servos. Por esta razón, para el movimiento de las articulaciones, se creó métodos que serán llamados cada vez que ingresen datos del tipo string. Además, está configurado el Arduino como modo cliente receptando la información que proviene del casco EEG. En la figura 48 se observa el diagrama de bloques del control del prototipo.

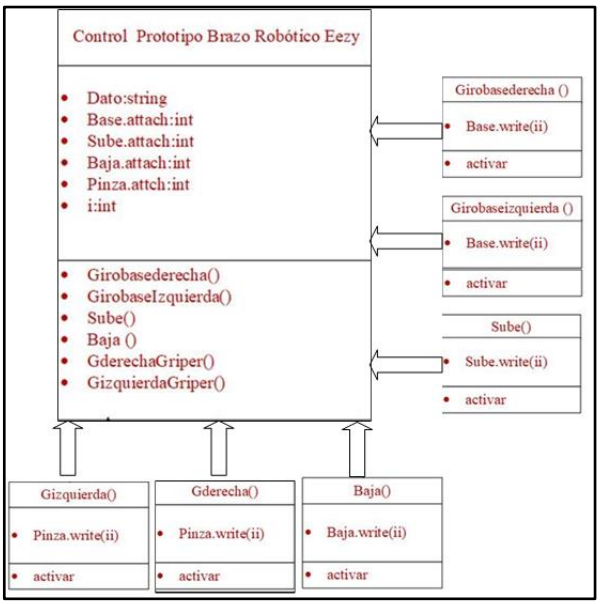


Figura 48: Diagrama de bloques del control del brazo prototipo

Elaborado por: El investigador

El control y desplazamiento del brazo robótico Eezy, ilustrada en la figura 49, permite al usuario manipular cada una de sus articulaciones mediante las señales electroencefalográficas que a través del sistema monitoreo y control envía pulsos al Arduino activando cada una los métodos permitiendo el movimiento del prototipo Eezy. En el anexo 9 se detalla la programación de las articulaciones del prototipo.

Figura 49: Implementación del sistema

Elaborado por: El investigador

Control del brazo robótico en Unity 3D

Ensamblaje del prototipo de brazo Robótico

El ensamblaje del prototipo se realizó en la ventana "Escena", los objetos para el ensamblaje del robot aparecen en la ventana "explorador", en la carpeta que previamente son guardados. Como se aprecia en la figura 50.

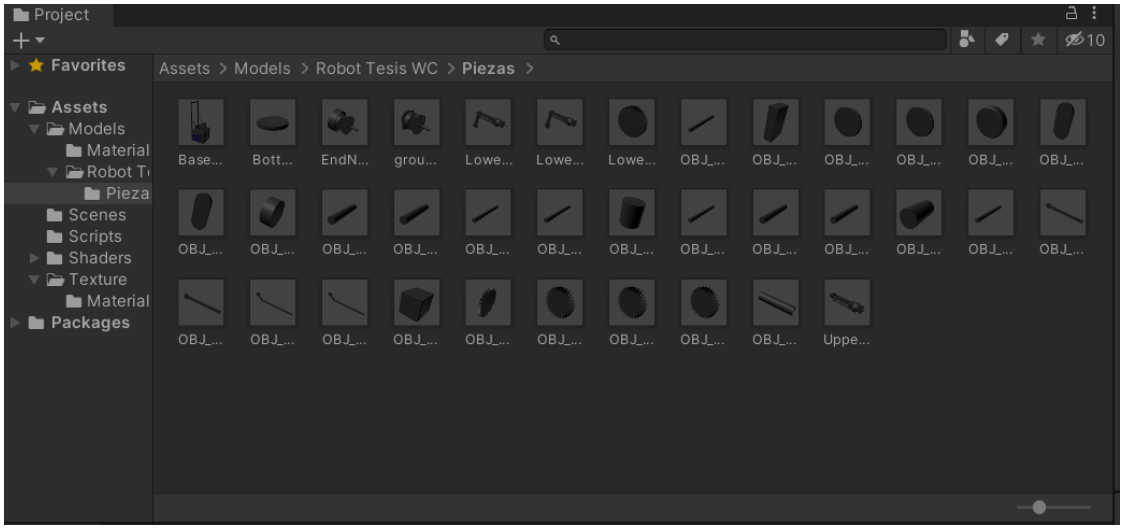


Figura 50: Piezas del brazo robótico.

Elaborado por: El investigador

Se arrastro todos los objetos necesarios hacia la ventana “Escena”, como se trabajó con objetos 3D, cada objeto fue ubicado en el espacio manteniendo su orientación con respecto al eje tridimensional de Unity, como se aprecia en la figura 51.

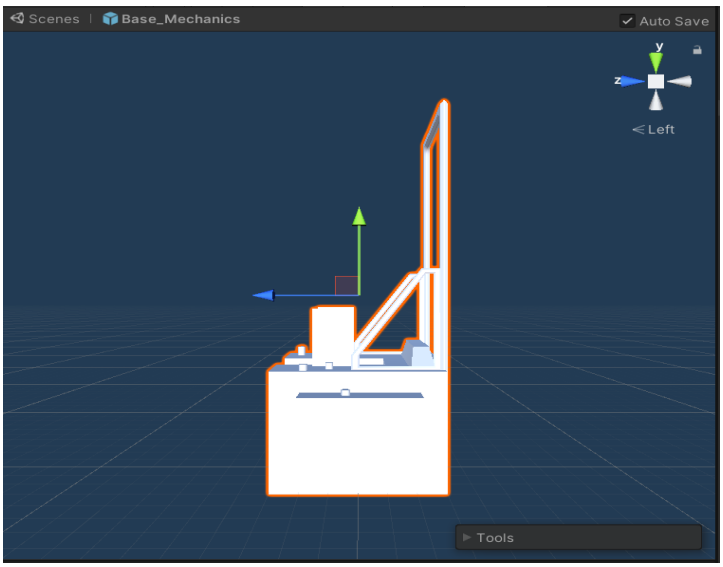


Figura 51: Pieza del brazo robótico dentro de la ventana Escena en Unity 3D.

Elaborado por: El investigador

Cada una de las piezas se ubicaron de manera de lograr el ensamble del brazo robótico, la ventana “Inspector” de Unity permite visualizar, añadir y quitar componentes al objeto como cuerpo rígido, colisionador, malla, etc. El ensamblaje del brazo robótico se logró de manera precisa mediante la utilización del componente “Transform”, que

proporciona la información de la posición, rotación, escala del objeto y también nos permite modificar estos parámetros.

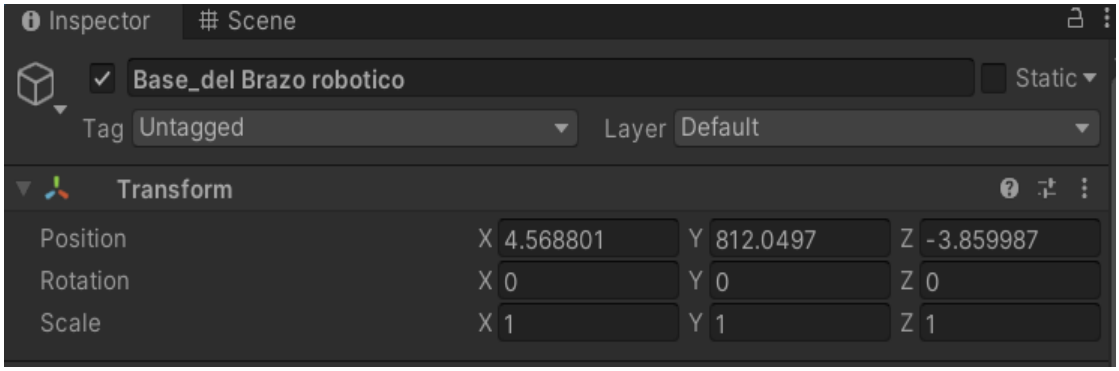


Figura 52: Ventana Inspector muestra la base del Brazo robótico

Elaborado por: El investigador

De esta forma se arma toda la estructura fija del brazo robótico, como se aprecia en la figura 53.

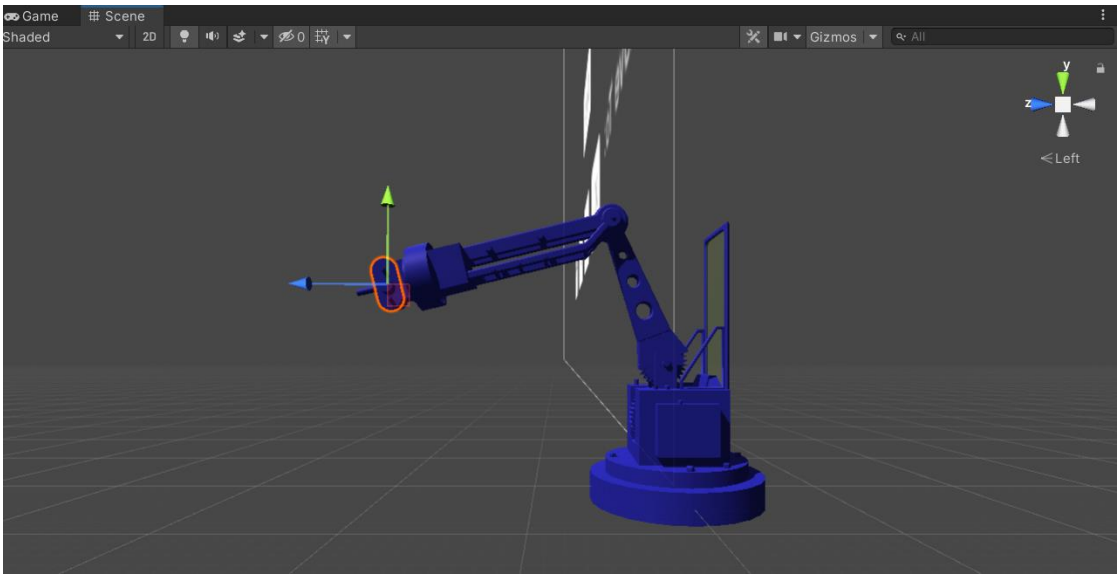


Figura 53: Brazo robótico ensamblada en Unity 3D

Elaborado por: El investigador

Cuando las piezas se encuentran en la posición que se desea para dar la forma al brazo robótico no se puede realizar ningún cambio de posición, ya que si se desea mover el robot se debería hacer el cambio a cada pieza.

Para añadir colores a las piezas del brazo robótico primero se crea un “material”, el cual por preferencia se debe guardar en una carpeta diferente para facilidad de uso las piezas. En la ventana inspector se encuentra los detalles del material creado y a su vez estos pueden ser modificados.

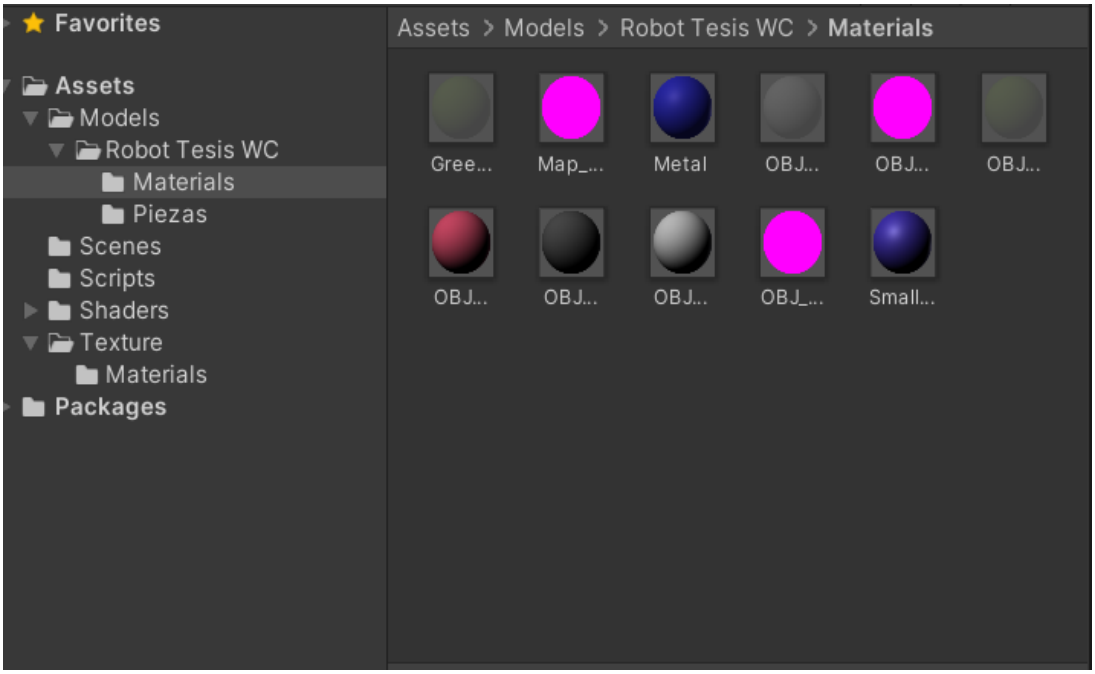


Figura 54: Mariales utilizados en el brazo robótico

Elaborado por: El investigador

Con el modelo ya en el espacio de trabajo se procede a darle características físicas y cinemáticas para el movimiento de cada una de las partes del robot, además se le agrega texturas y así darle el color característico. Finalmente se ha agregado un fondo para darle un aspecto agradable.

El software Unity 3D se instaló en una computadora con Windows 10, conectada a la red del sistema. El modelo 3D del prototipo del brazo se desarrolló con la finalidad de visualizar los movimientos que se ejecutan en el brazo robótico físico. El protocolo utilizado es el MQTT por sus ventajas: seguridad y fluidez de datos.

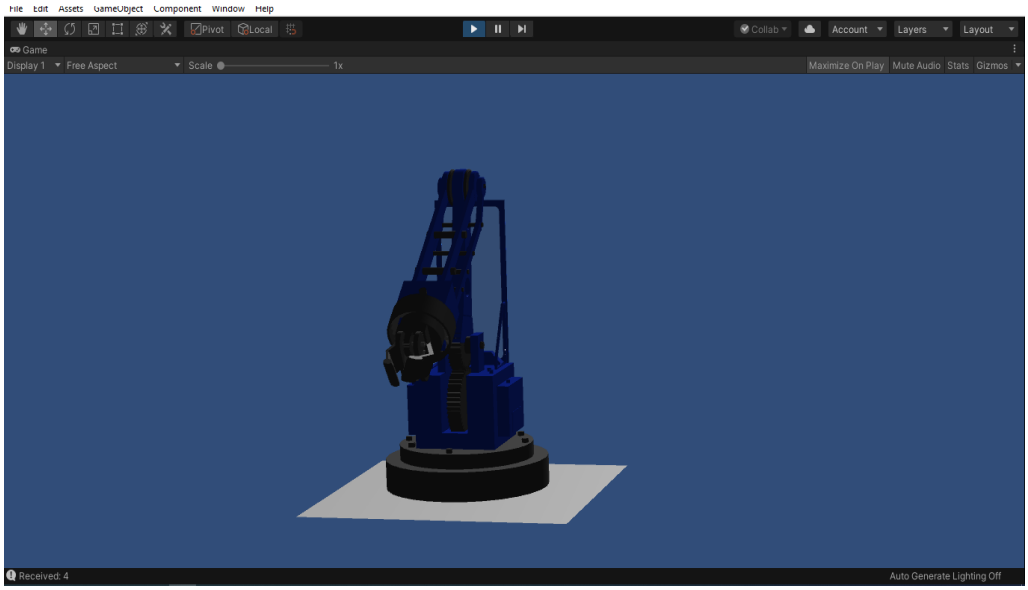


Figura 55: Simulación del brazo robótico

Elaborado por: El investigador

Para ayudar a la programación Unity provee un "Scripting API (Aplication Program Interface)" que se encuentra en la web de la documentación de Unity, esta sección de la documentación contiene detalles de la referencia de API que Unity provee. En la figura 56 se aprecia el esquema de la programación, el Script del brazo robótico es de una sola clase, que se encuentra conformado por la definición de las librerías, declaración de variables, recepción de datos, y seis rutinas principales para el movimiento del prototipo en el anexo 10 se aprecia la programación.

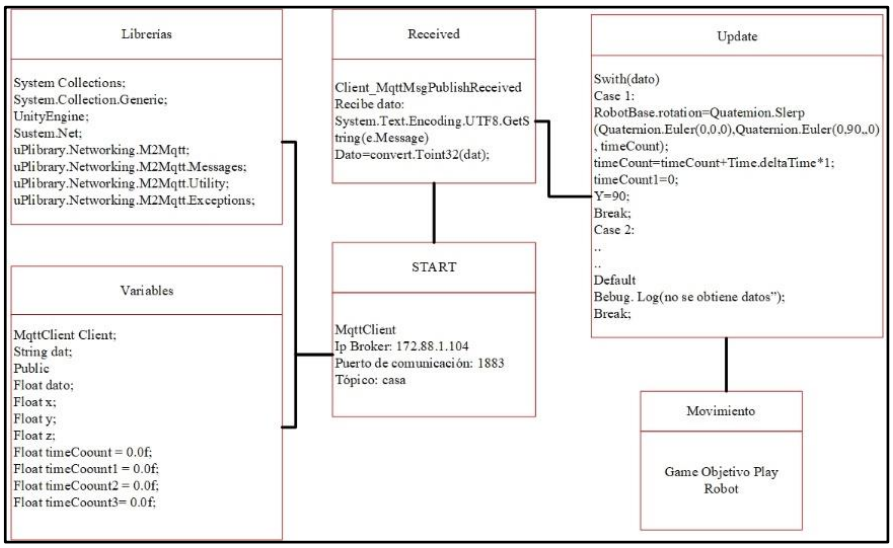


Figura 56: Diagrama conexión MQTT en Unity

Elaborado por: El investigador

Unity 3D brinda la posibilidad de comunicarse a través del protocolo MQTT, al recibir los datos de la interfaz gráfica de Matlab, además se puede visualizar las acciones que el prototipo físico que está realizando desde cualquier ordenador que se encuentre conectado a la red del sistema.

Control de un vehículo a escala

En la Figura 57 se observa la manipulación del prototipo de carro a control remoto, se utilizó la tarjeta ESP32 que tiene la ventaja de conectarse inalámbricamente a la red, mediante la configuración del protocolo MQTT como suscriptor, recepta los datos enviados desde el sistema, activando el integrado L293N y por consecuencia desplazándose el prototipo.

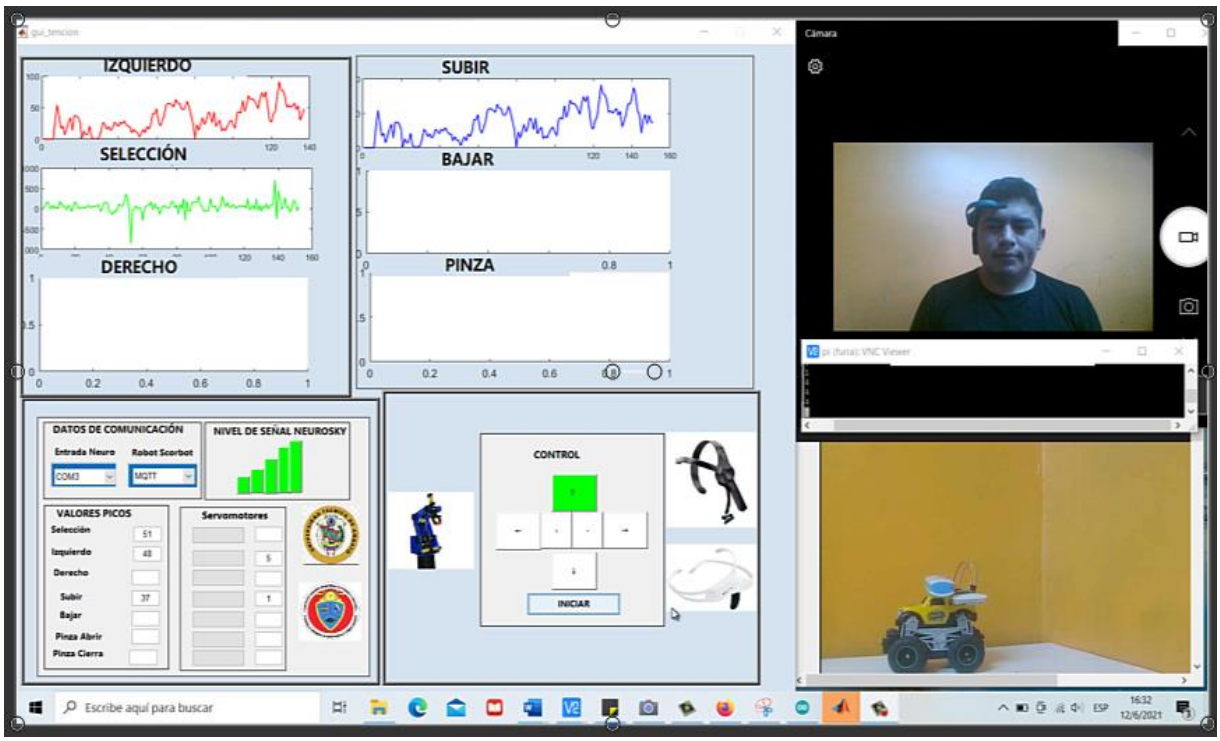


Figura 57: Control del carro a control remoto

Elaborado por: El investigador

3.2.6 Presupuesto del prototipo

Para determinar el costo total de la propuesta de investigación se deben considerar dos aspectos: el presupuesto de diseño y el presupuesto de ensamblaje del prototipo. Para el presupuesto de diseño se investigó el salario básico de un Ingeniero en Electrónica

determinado por el Ministerio de Trabajo correspondiente a 430.60 dólares mensuales. Considerando un promedio de 21 días laborables durante cada mes y aplicando la ecuación 1 se obtiene el salario diario.

$$\text{Salario}_{\text{diario}} = \frac{\text{Salario}_{\text{mensual}}}{\text{Dias}_{\text{Laborales}}} \quad (1)$$

$$\text{Salario}_{\text{diario}} = \frac{430.60}{21}$$

$$\text{Salario}_{\text{diario}} = 20.50$$

Es conocido que el día está constituido de 8 horas laborales, aplicando la ecuación 2 se obtiene la remuneración por hora de trabajo.

$$\text{Salario}_{\text{hora}} = \frac{\text{Salario}_{\text{diario}}}{\text{Horas}_{\text{Laborales}}} \quad (2)$$

$$\text{Salario}_{\text{diario}} = \frac{20.50}{8}$$

$$\text{Salario}_{\text{diario}} = 2.56$$

Se estiman 100 horas de investigación empleadas para el diseño, simulación y pruebas de funcionamiento; aplicando la ecuación 3 se obtiene el presupuesto de diseño del proyecto de investigación

$$\text{Presupuesto}_{\text{diseño}} = \text{Horas}_{\text{investigacion}} * \text{Salario}_{\text{horas}} \quad (3)$$

$$\text{Presupuesto}_{\text{diseño}} = 150 * 2.56$$

$$\text{Presupuesto}_{\text{diseño}} = 384 \text{ dolares}$$

El presupuesto para el ensamblaje del prototipo es importante, debido a que se utiliza equipos de hardware y software, y materiales de uso comercial, el costo final del diseño de la implementación del sistema es bajo, comparado con los sistemas que se encuentran en el mercado y que se utilizan en las empresas industriales. En la tabla 17 se muestra los costos de los equipos y materiales empleados para el prototipo del proyecto desarrollado.

Tabla 17: Presupuesto del proyecto

Presupuesto					
Items	Descripción	Unidad	Cantidad	Valor Unitario	Valor total
Elementos Electrónicos					
1	Esp32 NodeMCU	c/u	1	$15,00	$15,00
2	Raspberry Pi3 B	c/u	1	$60,00	$60,00
3	Arduino Mega	c/u	1	$10,00	$10,00
4	Brazo Robótico Eezy	c/u	1	$60,00	$60,00
5	Cables de conexión	c/u	1	$15,00	$15,00
6	Cables de red	c/u	4	$1,00	$4,00
7	Kit de elementos electrónicos	c/u	1	$25,00	$25,00
8	Integrados L298N	c/u	4	$4,20	$16,80
Equipos Electrónicos					
7	Casco Neuroskyn	c/u	1	$60,00	$60,00
8	Casco Emotiv insight	c/u	1	$322,00	$322,00
Materiales					
9	Estaño	c/u	1	$0,50	$0,50
10	otros materiales	c/u	1	$16,50	$16,50
				Subtotal	$604,80
			Mano de	obra el diseño	$384,00
				Total	**$988,80**

Elaborado por: El investigador

3.2.7 Análisis y Discusión de los Resultados

Temperatura de Raspberry

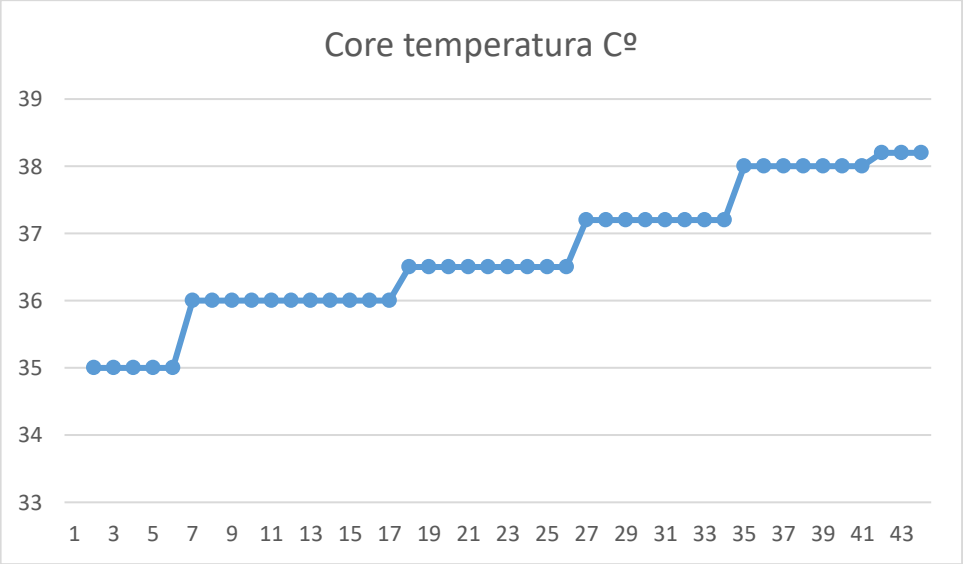


Figura 58: Temperatura

Elaborado por: El investigador

En la figura 58 se ilustra la imagen de la temperatura, durante los primeros 4 segundos oscila entre 35 y 36,5 grados centígrados, posterior a los 20s la temperatura oscila 37 a 38,9 grados centígrados. Una vez inicia los procesos y puesto en marcha la comunicación entre el Matlab, Unity y el Arduino la temperatura se estabiliza en 39 grados.

Los resultados obtenidos reflejan el uso de recursos de la tarjeta cuando el sistema de control se pone en acción, gráfica la tarjeta Raspberry se encontraba encendida pero aún no establecía comunicación con el sistema, en segunda instancia ya puesto en marcho la comunicación del sistema aumenta su temperatura progresivamente hasta estabilizarse 39 grados y conforme el tiempo avanza mantiene la temperatura garantizando que en ningún instante del tiempo la tarjeta llega a utilizar gran cantidad de recursos.

Resultados finales del proyecto

Para tener acceso al flujo de datos puros de los sensores EEG del casco emotiv insight, hay que contratar una suscripción a la empresa emotiv; sin embargo, los únicos datos que proporciona la empresa de forma gratuita son los movimientos de los sensores. Todo este procedimiento se da a través de una identificación única de Emotiv, que se otorga después del registro en el sitio web de emotiv.

Antes de usar el sistema, hay que humedecer bien todos los sensores y luego se tiene una hora para hacer las pruebas. Pasado ese tiempo los sensores se secan y esto conduce a resultados de prueba incorrectos. Las señales del cerebro humano inicialmente son receptadas por el auricular y enviados al ordenador mediante bluetooth. Luego, los datos se procesan a través del panel de control de emotiv y con este software se detecta el tipo de movimiento.

Después del reconocimiento de algunos de los tipos del movimiento, el software emokey traduce los resultados de la detección a los caracteres del teclado, que se envían en secuencia de comandos al panel de control del sistema y finalmente el robot realiza el movimiento deseado. El sistema de manipulación y monitorio necesita reconocer cada uno de los comandos que se envían al robot scorbot er 4u y a las

aplicaciones que se realizó en este proyecto anteriormente mencionado. Para ello en la tabla 18 se muestra los tiempos en segundos recolectados, se realizaron 20 peticiones de desplazamiento para cada uno de las condiciones de movimiento, utilizando el casco emotiv insigth.

Tabla 18: Tiempos de respuesta de reconocimiento de los comandos utilizando el Emotiv

Emotiv Insigth					
					
left	**rigth**	**rotate left**	**rotate rigth**	**pull**	**Blink**
2,3	2,5	2,6	3,8	1,2	1,6
2,1	2,3	2,2	2,5	2,6	2,8
2,4	2,2	2,3	2,8	2,4	2,4
2,8	2,6	2,4	2,6	2,6	2,6
2,6	2,1	2,6	2,4	2,7	2,4
2,4	2,1	2,6	2,3	2,9	2,9
2,3	2,3	2,6	2,7	2,4	2,1
2,7	2,7	2,1	2,9	2,3	2,9
2,9	2,2	2,4	2,1	2,1	2,9
2,1	2,6	2,4	2,3	2,6	2,3
2,3	2,7	2,1	2,9	2,3	2,2
2,9	2,9	2,8	2,2	2,6	2,4
2,2	2,8	2,4	2,5	2,8	2,5
2,5	2,9	2,3	2,2	2,9	2,1
2,2	2,5	2,7	2,5	2,8	2,3
2,5	2,7	2,4	2,4	2,2	2,7
2,4	2,4	2,8	2,5	2,1	2,3
2,5	2,4	2,7	2,3	2,9	2,8
2,2	2,8	2,4	2,5	2,8	2,5
2,1	2,2	2,4	2,3	2,4	2,4
2.42	**2.495**	**2.46**	**2.535**	**2.48**	**2.455**

Elaborado por: El investigador

El casco neuroskyn proporciona una librería liberada, el cual permite la adquisición de las señales puras del sensor, logrando así el filtrado y la manipulación de cada una de las señales en función de ciertas condiciones. En la tabla 19 se presenta la tabulación

de los datos de los tiempos de reconocimiento de los comandos por el sistema, utilizando el casco neuroskyn.

Tabla 19: Tiempos de respuesta de reconocimiento utilizando el Neuroskyn

Neuroskyn					
Atención 60 uV-80 uV	Atención 80 uV-100 uV	Meditación 80 uV– 100 uV	Meditación 60 uV-80 uV	Atención 40 uV – 60 uV	Meditación 40 uV – 60 uV
					
6,1	5,6	4,9	5,9	3,9	4,8
6,3	6,6	4,8	5,3	3,8	4,9
6,9	5,7	4,6	7,9	3,7	4,3
5,8	5,8	4,7	6,9	3,7	4,7
7,1	5,9	4,6	5,3	3,8	4,2
5,9	5,3	4,9	5,8	3,8	4,8
7,9	5,6	4,6	5,8	3,8	4,2
6,3	4,9	4,1	5,9	3,2	4,7
5,8	5,3	4,7	5,7	4,1	4,7
5,9	5,7	4,3	5,8	4,2	4,1
7,1	5,7	4,9	5,6	4,9	4,2
7,2	5,9	3,9	5,7	4,7	3,9
7,8	5,9	4,8	5,8	4,8	3,2
7,9	5,7	4,9	5,9	4,8	3,7
7,8	5,3	3,8	5,7	4,1	4,7
7,8	5,3	4,7	5,8	4,9	3,8
7,9	5,7	4,8	5,8	4,1	3,2
7,8	5,4	4,9	5,7	4,7	3,7
7,9	5,7	4,9	5,9	4,3	3,7
7,4	6,0	4,2	4,5	4,7	4,5
7.01	**5.65**	**4.6**	**5.835**	**4.2**	**4.2**

Elaborado por: El investigador

En la tabla 20 se presenta las tabulaciones de los tiempos de reconocimiento de los comandos por el sistema tanto para el casco neuroskyn como para el emotiv, para cada aplicación.

Tala 20: Cuadro comparativo del tiempo de reconocimiento de los comandos

	Robot Scorbot 		Silla de Ruedas 		Carro a control 		Brazo robotic 		Unity 3D 	
										
Izquierda 	7,01	2,42	5,8	2,41	5,2	2,3	5,3	2,2	4,2	2,1
Derecha 	5,65	2,495	6,5	2,6	4,2	2	4,2	2,1	4,3	2,3
Subir 	4,6	2,46	5,1	4,2	5,6	2,4	5,7	3,2	4,2	2,3
Bajar 	5,835	2,535	4,9	3,9	2,4	5,3	4,3	3,5	3,9	2,1
Abrir 	4,2	2,48	5,1	2,9	5,6	5,8	4,7	3,7	3,8	2,8
Cerrar 	4,2	2,455	5,6	2,3	7,2	4,2	5,3	3,8	5,1	2,3

Elaborado por: El investigador

En la Figura 59 se observa la gráfica de los tiempos de respuesta que le toma al sistema en reconocer los comandos de cada una de las aplicaciones realizadas utilizando los dos cascos emotiv y el neuroskyn respectivamente. El tiempo máximo de respuesta de las condiciones de movimiento esta entre 2.535 y 2.490 segundos y el mínimo es de 2.42 estos valores son inferiores al comparar con los datos del casco neuroskyn que tiene un tiempo de respuesta máxima de 7.01 segundos y un tiempo mínimo inferior a 4.20 segundos, al manipular el robot scorbot.

Para el control de la silla de ruedas tiene un tiempo de respuesta máximo de 4.2s y un tiempo mínimo inferior a los 2.3s, teniendo mayor concentración de tiempos de respuesta en un rango 2.41 a 2.9s a diferencia de los tiempos de respuesta al utilizar el casco neuroskyn que tiene un tiempo de respuesta máximo de 6.5 s y un tiempo mínimo de 4.9 s. Para el control del brazo robótico Eezy los tiempos de respuesta mínimos son 2.1s, con una variación que llega hasta 3.8s; teniendo la mayor concentración de tiempos de respuesta en el rango comprendido entre los 2.2s, a 3.7s. El tiempo máximo de respuesta 5.7s utilizando el casco neuroskyn un tiempo mínimo de respuesta de 4.2s. Para el tiempo de respuesta de la simulación en unity 3D el rango

de tiempos se extiende desde 2.1s a 2.8s, a diferencia del auricular neuroskyn que el rango comprendido entre 4.2s a 5.1s.

Figura 59: Tiempos de respuesta de reconocimientos de los comandos por el sistema

Elaborado por: El investigador

CAPÍTULO IV
CONCLUSIONES Y RECOMENDACIONES

4.1. Conclusiones

- El análisis de los sistemas de control inteligente basados en el uso de las señales electroencefalográficas está conformado por etapas que se clasifican en la adquisición, acondicionamiento, procesamiento de las señales y ejecución de las aplicaciones, constituyendo elementos del sistema BCI (Interfaz cerebro computadora), tomando en cuenta los dispositivos EEG por sus características técnicas, que permite la adquisición no invasiva de electroencefalogramas.

- El sistema de control y monitoreo se conecta al identificador de la conexión al puerto seria com3 asignado por el mindwave neuroskyn adquiriendo la información enviada desde el sensor Neuroskyn TGAM EEG, obteniendo las señales sin procesar que mediante funciones y algoritmos se filtra y procesa. Con la finalidad de generar datos de envió, que a través del protocolo MQTT, utilizando el estándar 802.11n que perteneciente a la red inalámbrica, establece una conexión con los periféricos de las aplicaciones como es el brazo robótico Scorbot ER 4U, una silla de ruedas, un prototipo de brazo robótico Eezy, una simulación en unity 3d y un carro a escala a control remoto, demostrando así el correcto funcionamiento de cada uno de los procesos descritos anteriormente y mostrando tener la estabilidad que se espera en todo el sistema.

- A demás se utilizó el auricular Emotiv Insignt, y las aplicaciones Emokey como una alternativa para obtener señales EEG de bajo costo para el control del brazo robótico y las aplicaciones, por lo que es necesario hacer uso de la librería emotiv sdk (edk.dll), debido a que contiene las API (interfaz de programación de aplicaciones) permitiendo la comunicación entre dos aplicaciones de software, como la conexión con la interfaz gráfica del sistema de control y monitoreo

- Dentro del campo de las tecnologías se cuenta con varios tipos de software de procesamiento, pero para la manipulación de señales EEG se utilizó el Matlab creando un programa principal que gestiona los datos de los dispositivos electroencefalográficas, la cual brinda graficas de las señales captadas y datos

procesados. Como controlador del brazo robótico Scorbot ER 4U se empleó la tarjeta Rasberry Pi 3 B+, pues se adapta perfectamente a la gran mayoría de protocolos de comunicación pudiendo así interactuar con diferentes plataformas y para el control de las aplicaciones como es la silla de ruedas, el carro a control se utilizó la controladora Esp32 a diferencia el brazo robotico Eezy que manejo el Arduino mega.

- La implementación del sistema de manipulación y monitoreo del robot scorbot mediante las señales electroencefalográficas (EEG) resulta eficiente debido a los tiempos de respuesta de reconocimiento de los comandos por el sistema de manipulación resultan inferiores a 5.8s; con una concentración del 80% en el tiempo de respuesta de 2.3s. utilizando el casco emotiv insight, y una concentración del 85% en el tiempo de respuesta de reconocimiento utilizando casco Neuroskyn de 5.6s.

4.2. Recomendaciones

- En el desarrollo del sistema de control, se recomienda la creación de métodos para cada una de las etapas, para que el sistema pueda trabajar con fluidez y correcto funcionamiento.

- Al utilizar los cascos EEG para enviar datos a dispositivos electrónicos se recomienda investigar todo lo referente a la comunicación entre dispositivos protocolos, estándares, librerías, tecnologías e interfaces, etc. puesto que en este proyecto se unifico todo el sistema.

- Para obtener los datos del auricular Emotiv Insigth se recomienda configurar la interfaz de programación de aplicaciones, ya que integra una librería vinculada y se accede a ella a través de un conjunto de funcione (Api).

- Para disminuir los tiempos de reconocimiento de los caracteres por el sistema de control y monitoreo, enviados por las diademas de señales EEG es recomendable realizar entrenamientos repetitivos para familiarizarse con la generación de las señales, pues de estos factores depende la velocidad de respuesta del sistema.

REFERENCIAS BIBLIOGRAFICAS

[1] M. AlAbboudi, M. Majed, F. Hassan, and A. B. Nassif, «EEG wheelchair for people of determination,» 2020. [En linea]. Available : https://ieeexplore.ieee.org/document/9118340. [Ultimo acceso: 13 07 2021].

[2] I. N. Zamora, D. S. Benitez, and M. S. Navarro, «On the Use of the EMOTIV Cortex API to Control a Robotic Arm Using Raw EEG Signals Acquired from the EMOTIV Insight NeuroHeadset,» 12 2019. [En linea]. Available : https://ieeexplore.ieee.org/document/8987541. [Ultimo acceso: 13 07 2021].

[3] Y. Li, F. Zhang, and Y. Yang, "Smart House Control System Controlled by Brainwave," *10* 2019 Available : https://ieeexplore.ieee.org/document/8527397. [Ultimo acceso: 13 07 2021].

[4] Edwin and Vaca, «Prototipo de prótesis de un brazo con 12 GDL controlada mediante ondas cerebrales». 2017. [En linea]. Available : https://repositorio.uta.edu.ec/handle/123456789/24670. [Ultimo acceso: 13 07 2021].

[5] Y. Yordanov, G. Tsenov, and V. Mladenov, «Humanoid robot control with EEG brainwaves, » 2 2017. [En linea]. Available : https://ieeexplore.ieee.org/document/8095083. [Ultimo acceso: 13 07 2021].

[6] S. K. Swee and L. Z. You, «Fast Fourier analysis and EEG classification brainwave controlled wheelchair,» 23 2016. [En linea]. Available : https://ieeexplore.ieee.org/document/7784344. [Ultimo acceso: 13 07 2021].

[7] A. Sakaguchi and T. Takimoto, «Development of support system for multi-rotor helicopter operations using electroencephalograph,» 2014. [En linea]. Available : https://ieeexplore.ieee.org/document/6988046. [Ultimo acceso: 13 07 2021].

[8] D. P. Mital and G. W. Leng, "A voice-activated robot with artificial

intelligence," 1988. [En linea]. Available: https://ieeexplore.ieee.org/document/239170 [Ultimo acceso: 05 02 2021].

[9] E. Ivorra, M. Ortega, M. Alcaniz, and N. Garcia-Aracil, "Multimodal Computer Vision Framework for Human Assistive Robotics," [En linea]. Available: https://ieeexplore.ieee.org/document/8428330 [Ultimo acceso: 05 02 2021].

[10] R. Suescun. "Robots en América Latina" [En linea]. Available: https://blogs.iadb.org/gestion-fiscal/es/robots-en-america-latina-cuantos-son-donde-estan-y-cuanto-tributan/ [Ultimo acceso: 05 02 2021].

[11] R. Camana, "La robótica industrial en el Ecuador." https://robertocamana.wordpress.com/2014/12/03/robotica-industrial-ecuador/ [Ultimo acceso: 05 02 2021].

[12] A. Salazar, "Propuesta de prototipo de robot aspiradora de bajo costo y alta tecnología aplicado a procesos de limpieza de baja escala." [En linea]. Available: https://revistas.uteq.edu.ec/index.php/ingenio/article/view/358/411 [Ultimo acceso: 3 02 2021].

[13] E. R. Miranda, "*Guide to Brain- Computer Music Interfacing*.," 2014. [En linea]. Available: https://www.springer.com/gp/book/9781447165835. [Ultimo acceso: 01 10 2021]..

[14] G. Mosquera and S. Daniel, "Adquisición de señales electroencefalográficas para el movimiento de un prototipo de silla de ruedas en un sistema BCI," *Univ. Politec. Saleciana*, vol. 1, no. 1, p. 83, 2012, [Online]. Available: https://dspace.ups.edu.ec/bitstream/123456789/3212/1/UPS-CT002509.pdf. [Ultimo acceso: 13 07 2021].

[15] "Cuadros sinópticos y mapas conceptuales sobre las neuronas | Cuadro

Comparativo.” https://cuadrocomparativo.org/cuadros-sinopticos-sobre-las-neuronas/ [Ultimo acceso: 10 01 2021].

[16] L. García, “Control del robot IRB120 mediante el casco de electroencefalografía Neurosky Mindwave,” 2017, [Online]. Available: https://ebuah.uah.es/dspace/handle/10017/30246. [Ultimo acceso: 13 07 2021].

[17] J. Gutiérrez-martínez, J. Cantillo-negrete, R. I. Cariño-escobar, and D. Elías-viñas, “Los sistemas de interfaz cerebro-computadora: una herramienta para apoyar la rehabilitación de pacientes con discapacidad motora,” 2013, [Online]. Available: http://new.medigraphic.com/cgi-bin/resumen.cgi?IDARTICULO=44426. [Ultimo acceso: 13 07 2021].

[18] E. Guacho, «Implementación de un sistema de control para el manipulador mitsubishi rv-2aj, mediante ondas cerebrales empleando el sensor emotiv insight,» 2018 [En linea]. Available : https://dspace.ups.edu.ec/handle/123456789/15886. [Ultimo acceso: 13 07 2021].

[19] “Unobrain: un gimnasio cerebral con un casco que «lee la mente»,” 2021. [En linea]. Available: https://www.microsiervos.com/archivo/gadgets/unobrain-gimnasio-mental.html. [Ultimo acceso: 01 10 2021].

[20] E. De Ingeniería, E. N. Electrónica, J. Jaime, T. Tinoco, O. Fernando, and S. Fernández, «Diseño e implementación de una red wsn, para el control de eventos físicos a traves de señales cerebrales como ayuda a personas con capacidades físicas limitadas, » 2017. [Online]. Available: http://dspace.espoch.edu.ec/bitstream/123456789/7951/1/98T00169.pdf. [Ultimo acceso: 01 10 2021].

[21] “Robótica: una década de transformaciones | OpenMind.” [En linea]. Available: https://www.bbvaopenmind.com/articulos/robotica-una-decada-de-

transformaciones.html. [Ultimo acceso: 04 2 2021].

[22] Juan A. Alonso, Santiago Blanco A., Santiago Blanco S. “Robots industriales,” [Online].Available:http://platea.pntic.mec.es/vgonzale/cyr_0708/archivos/_15/Tema_5.4.htm [Ultimo acceso: 04 2 2021].

[23] T. Versatile and T. Robot, “SCORBOT-ER 4u,” pp. 4–5. [En linea]. Available: https://www.intelitek.html. [Ultimo acceso: 04 2 2021].

[24] “Raspberry Pi.” [En linea]. Available: https://raspberrypi.cl/que-es-raspberry.html. [Ultimo acceso: 04 1 2021].

[25] T. E. Industrial, “Estimación De Posturas Humanas Para Un Robot Omnidireccional Kuka Youbot “,” 2019.

[26] P.Bonilla., “Diseño de sisemas de control industrial de robots basados en industria 4.0,” *J. Chem. Inf. Model.*, vol. 21, no. 1, pp. 1–9, 2020, [Online]. Available: https://repositorio.uta.edu.ec/handle/123456789/29952 [Ultimo acceso: 13 7 2021].

[27] P. Cuspha., “Comunicación y virtualización de procesos industriales basados en industria 4.0,” 2020. [Online]. Available: http://repo.uta.edu.ec/bitstream/handle/123456789/5301/Mg.DCEv.Ed.1859.pdf?sequence=3. [Ultimo acceso: 04 1 2021].

[28] McGraw-Hill, “Protocolo TCP/IP,” *May 25, 2012*, p. 24, 2012, [Online]. Available: http://assets.mheducation.es/bcv/guide/capitulo/8448199766.pdf. [Ultimo acceso: 04 1 2021].

[29] C. A. espinoza Chaparro and Santiago Grisales Naranjo Hector Fabio Mayor DIez, “No Title.” p. 111, 1995, [Online]. Available: https://core.ac.uk/download/pdf/229157261.pdf. [Ultimo acceso: 04 1 2021].

[30] “Conozca MQTT – IBM Developer.” [Online]. Available: https://developer.ibm.com/es/articles/iot-mqtt-why-good-for-iot/ [Ultimo acceso: 10 1 2021].

[31] R. A. Light, “Mosquitto: server and client implementation of the MQTT protocol,” [Online]. Available: https://www.researchgate.net/publication/317210457_Mosquitto_server_and_client_implementation_of_the_MQTT_protocol . [Ultimo acceso: 10 1 2021].

ANEXOS

Anexo 1: Instalación del casco Neurosky

Para la instalación del casco neuroskyn se descargó WindWave Mobile, este software es propia del casco, se instaló con la finalidad de adquirir la librería ThinkGear Connector, librería que va junto a los archivos de procesamiento del matlab.

Figura 60: Librerías ThinkGear

Elaborado por: El investigador

A continuación, se descarga los archivos .h necesarios para establecer comunicación entre el casco y el Matlab.

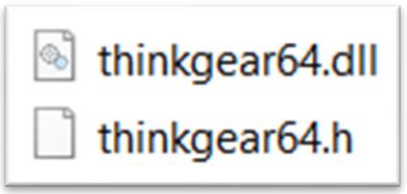


Figura 61: Archivos h y archivos dll

Elaborado por: El investigador

Hecho esto se copia los archivos descargados y la librería en una sola carpeta, posteriormente se verifica el puerto de configura como se ve en la figura 56.

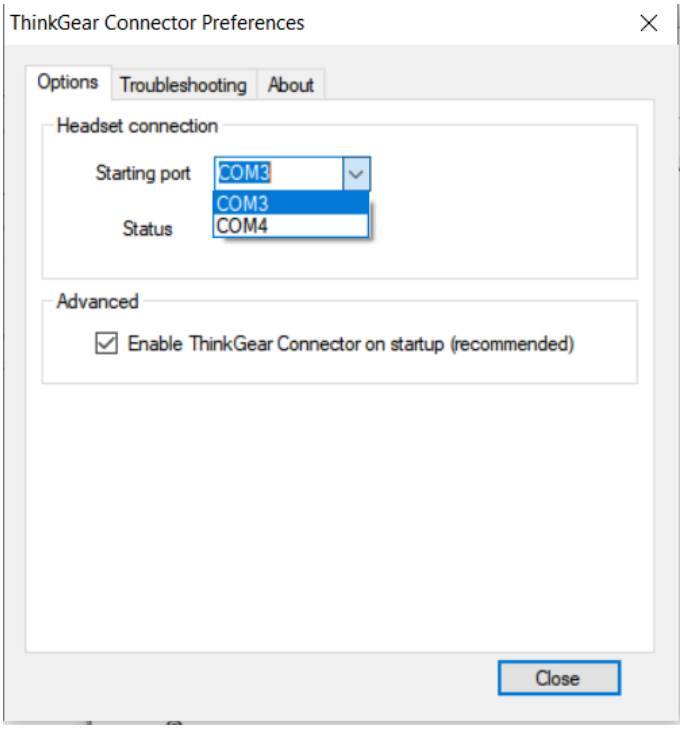


Figura 62: Puertos de configuración de entrada y salida com

Elaborado por: El investigador

Una vez identificado el puerto de ingreso, se utilizó el puerto para identificarlo en el programa .m de Matlab.

Anexo 2: Código de la señal de atención y meditación

```
clear all
close all
data = zeros(1,256);    % Asignar búfer
portnum1 = 3;
% Selección del puerto com
comPortName1 = sprintf('\\\\.\\COM%d', portnum1);
% Selección de la velocidad de los baudios para usar TG_Connect () y
TG_SetBaudrate ().
TG_BAUD_57600 =      57600;
% Formato de datos para usar con TG_Connect() and TG_SetDataFormat().
TG_STREAM_PACKETS =     0;
% Tipo de datos que se pueden solicitar desde TG_GetValue ().
TG_DATA_RAW =         4;
%Cargar la librería thinkgear dll
loadlibrary('thinkgear64.dll','thinkgear64.h');
fprintf('thinkgear64.dll loaded\n');
%Obtener la versión dll
dllVersion = calllib('thinkgear64', 'TG_GetDriverVersion');
fprintf('ThinkGear DLL version: %d\n', dllVersion );
 % identificador de conexión para  ThinkGear
connectionId1 = calllib('thinkgear64', 'TG_GetNewConnectionId');
if ( connectionId1 < 0 )
error( sprintf( 'ERROR:TG_GetNewConnectionId()returned %d.\n', connectionId1));
end;
% Establece y abre archivos de registros de flujo (bytes sin procesar) para la
conexión
errCode = calllib('thinkgear64', 'TG_SetStreamLog', connectionId1, 'streamLog.txt' );
if( errCode < 0 )
 error( sprintf( 'ERROR: TG_SetStreamLog() returned %d.\n', errCode ) );
end;
% Establece y abre archivo de registro de datos (valores ThinkGear) para la conexión
errCode = calllib('thinkgear64', 'TG_SetDataLog', connectionId1, 'dataLog.txt' );
if( errCode < 0 )
error( sprintf( 'ERROR: TG_SetDataLog() returned %d.\n', errCode ) );
end;
 % Intenta conectar al identificador de la conexión al puerto serie “COM3”
errCode = calllib('thinkgear64', 'TG_Connect',
connectionId1,comPortName1,TG_BAUD_57600,TG_STREAM_PACKETS );
if ( errCode < 0 )
error( sprintf( 'ERROR: TG_Connect() returned %d.\n', errCode ) );
end
 fprintf( 'Connected.  Reading Packets...\n' );
 %
%registra datos
j = 0;
i = 0;
while (i < 10240) % bucle durante 20 segundos
if (calllib('thinkgear64','TG_ReadPackets',connectionId1,1) == 1)  %leer
```

```
if (calllib('thinkgear64','TG_GetValueStatus',connectionId1,TG_DATA_RAW) ~=
0)   % si se ha actualizado RAW
j = j + 1;
i = i + 1;
data(j) = calllib('thinkgear64','TG_GetValue',connectionId1,TG_DATA_RAW);
end
end
if (j == 256)
plotRAW(data); % trazar los datos, actualizar cada .5 segundos (256 puntos)
j = 0;
end
end
% desconectar
calllib('thinkgear64', 'TG_FreeConnection', connectionId1 );
```

Descripción

En la primera parte del código se selecciona el puerto de comunicación del casco, la velocidad en baudios para usar la librería thinkgear dll , posteriormente se escoge el tipo de dato que solicita la librería, como siguiente paso se carga la librería verifica e identifica la conexión, establece los archivos de registro de flujos para la conexión registra los datos e ingresa a un bucle, lee los datos proporcionados por la señal TG_DATA_RAW y grafica las señales correspondientes a la meditación y atención.

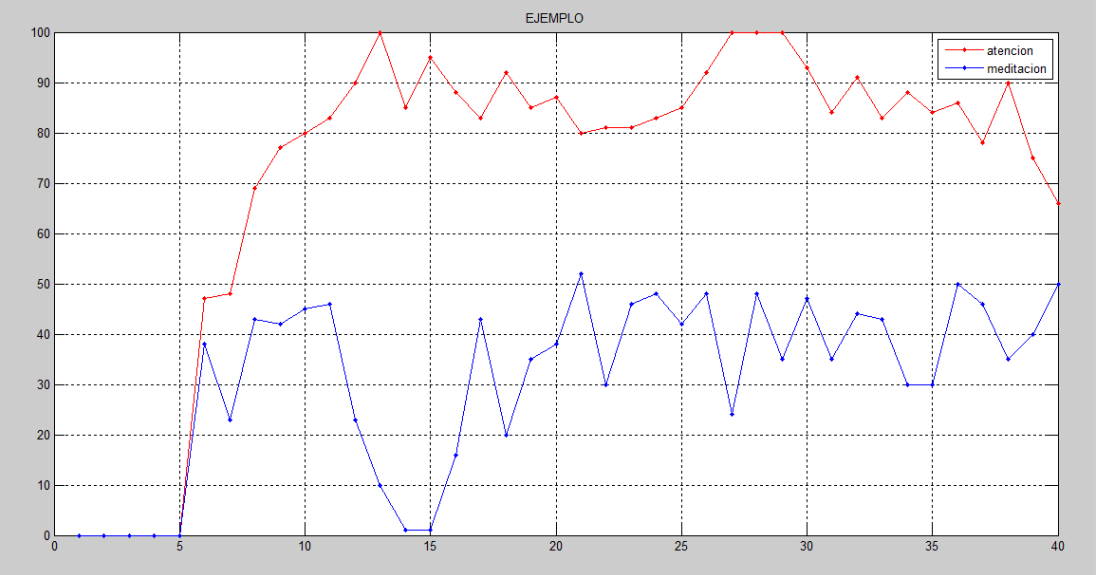


Figura 63: Señal de atención y meditación

Elaborado por: El investigador

Anexo 3: Código de la señal de Raw y Parpadeo

```
%Limpiar la pantalla
clc;
%Variables claras
clear all;
%Cierre de figuras
close all;
a=imread('blink.JPG');
%Asignar buffer
data_blink = zeros(1,256);
%Comport Selection
portnum1 = 3;
%selección del puerto para la comunicación del asco
comPortName1 = sprintf('\\\\.\\COM%d', portnum1);
% Selección de la velocidad de los baudios para usar TG_Connect () y
TG_SetBaudrate ().
TG_BAUD_115200 = 115200;
% Formato de datos para usar con TG_Connect () y TG_SetDataFormat ().
TG_STREAM_PACKETS = 0;
% Tipo de datos que devuelve TG_GetValue().
TG_DATA_BLINK_STRENGTH = 37;
% cargar thinkgear64 dll
loadlibrary('thinkgear64.dll');
%Mostrar la venta de comandos
fprintf('thinkgear64.dll loaded\n');
%get dll version
dllVersion = calllib('thinkgear64', 'TG_GetDriverVersion');
% Obteniendo un identificador de ID de conexión a ThinkGear
fprintf('thinkgear64 DLL version: %d\n', dllVersion );
% Obteniendo un identificador de identificación de conexión para ThinkGear
connectionId1 = calllib('thinkgear64', 'TG_GetNewConnectionId');
if ( connectionId1 < 0 )
error( sprintf( 'ERROR: TG_GetNewConnectionId() returned %d.\n', connectionId1 )
);
end;
% conectar el identificador de conexión al puerto serie “COM3”
errCode = calllib('thinkgear64', 'TG_Connect',
connectionId1,comPortName1,TG_BAUD_115200,TG_STREAM_PACKETS );
if ( errCode < 0 )
error( sprintf( 'ERROR: TG_Connect() returned %d.\n', errCode ) );
end
fprintf( ‘Connected. Reading Packets...\n’ ); % imprime los  carácter es leyendo
paquetes
if(calllib('thinkgear64','TG_EnableBlinkDetection',connectionId1,1)==0)
disp('blinkdetectenabled');
end
i=0;
j=0;
%Mostrar la venta de comandos
```

```
disp('Reading Brainwaves');
while i < 20
if (calllib('thinkgear64','TG_ReadPackets',connectionId1,1) == 1) %Si Lee el paquete
if
(calllib('thinkgear64','TG_GetValueStatus',connectionId1,TG_DATA_BLINK_STR
ENGTH) ~= 0)
j = j + 1;
i = i + 1;
%Lee los valores de atención de los paquetes
data_blink(j) =
calllib('thinkgear64','TG_GetValue',connectionId1,TG_DATA_BLINK_STRENGT
H );
%Mostrar la venta de comandos
disp(data_blink(j));
%Graficar las señales en axe
close all;
end
end
end
%Mostrar la venta de comandos
disp('Loop Completed')
%Desconecta los puertos de comunicaciones
calllib('thinkgear64', 'TG_FreeConnection', connectionId1 );
```

Descripción

En la primera parte del código selecciona el puerto de comunicación del casco, la velocidad en baudios para usar la librería thinkgear dll , posteriormente se escoge el tipo de dato que solicita la librería, como siguiente paso carga la librería verifica e identifica la conexión, establece los archivos de registro de flujos para la conexión registra los datos e ingresa a un bucle, lee los datos proporcionados por la señal TG_DATA_RAW y TG_DATA_BLINK_STRENGT, como se puede observar en la figura 58 se grafica las señales correspondientes al parpadeo y a la señal original RAW.

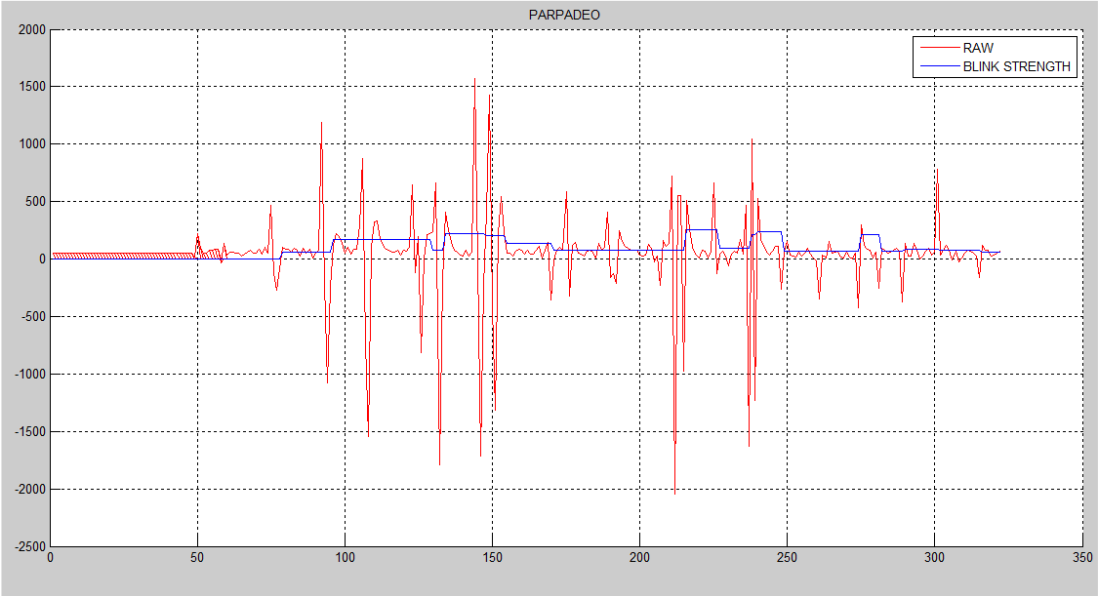


Figura 64: Señal de Raw y parpadeo

Elaborado por: El investigador

Anexo 4: Código para interpretación de caracteres de Emotiv Emokey para interpretación gráfica de Matlab.

```
function Tx_send__Callback(hObject, eventdata, handles)
% Valor enviado por el recuadro
TxText = get (handles.Tx_send, 'String');
fprintf(handles.serConn, TxText);
% Envía el contenido de la columna a la variable
currList = get(handles.history_box, 'String');
% Obtener la longitud de la columna
set(handles.history_box, 'Value', length(currList)+1;
%obtener contenido de columna
set(hObject, 'String', ' ' );
% comparar los caracteres
if TxText == ' a'
% ejecución del método
baseiz
end;
```

Anexo 5: Código unificado del sistema de monitoreo y control

```
function pushbutton1_Callback(hObject, eventdata, handles)
TG_DATA_ATTENTION = 2;
TG_DATA_MEDITATION=3;
TG_DATA_RAW=4;
TG_DATA_BLINK_STRENGTH=37;
data_att = zeros(40,3);
aten=zeros(30,1); %hemos recogido 40 datos
medi=zeros(30,1);
%Selección del puerto
portnum1 = 3;
selección del puerto para la comunicacion del asco
comPortName1 = sprintf('\\\\.\\COM%d', portnum1);
% Baud rate for use with TG_Connect() and TG_SetBaudrate().
TG_BAUD_57600 = 57600;
% Formato de datos para usar con TG_Connect () y TG_SetDataFormat ().
TG_STREAM_PACKETS = 0;
% Tipo de datos que se pueden solicitar desde TG_GetValue ().
TG_DATA_ATTENTION = 2;
%cargando thinkgear dll
loadlibrary('thinkgear64.dll');
%Mostrar la venta de comandos
fprintf('thinkgear64.dll.dll loaded\n');
%get dll version
dllVersion = calllib('thinkgear64', 'TG_GetDriverVersion');
%Mostrar la venta de comandos
fprintf('thinkgear64.dll DLL version: %d\n', dllVersion );
% Get a connection ID handle to ThinkGear
connectionId1 = calllib('thinkgear64', 'TG_GetNewConnectionId');
if ( connectionId1 < 0 )
error( sprintf( 'ERROR: TG_GetNewConnectionId() returned %d.\n', connectionId1 ) );
end;
% Attempt to connect the connection ID handle to serial port “COM3”
errCode = calllib('thinkgear64', 'TG_Connect',
connectionId1,comPortName1,TG_BAUD_57600,TG_STREAM_PACKETS );
if ( errCode < 0 )
%error( sprintf( 'ERROR: TG_Connect() returned %d.\n', errCode ) );
end
fprintf( ‘Connected. Reading Packets...\n’ ); % imprime los  carácter es leyendo paquetes
i=0;
j=0;
aux=0;
com=0; Asignada el valor 0 a la variable com
com1=0; Asignada el valor 0 a la variable com1
com2=0; Asignada el valor 0 a la variable com2
com3=0; Asignada el valor 0 a la variable com3
com4=0; Asignada el valor 0 a la variable com4
com5=0; Asignada el valor 0 a la variable com5
%Mostrar la venta de comandos
disp('Reading Brainwaves');
atencion=0;
meditacion=0;
t = timer('TimerFcn', 'stat=false; disp("FIN")','StartDelay',40);
```

```
%errCode = calllib('ThinkGear', 'TG_EnableAutoRead', connectionId1,-1);
start(t)
stat=true;
while (stat==true)
if (calllib('thinkgear64','TG_ReadPackets',connectionId1,1) == 1) %Si Lee el paquete
if (calllib('thinkgear64','TG_GetValueStatus',connectionId1,TG_DATA_ATTENTION ) ~=
0)
j = j + 1;
i = i + 1;
%Lee los valores de atención de los paquetes
data_att(j,1)=calllib('thinkgear64','TG_GetValue',connectionId1,TG_DATA_ATTENTION);
set(handles.pushbutton8,'backgroundcolor','g'); % Asigna el color verde al boton
set(handles.pushbutton12,'backgroundcolor','g'); % Asigna el color verde al boton
set(handles.pushbutton13,'backgroundcolor','g'); % Asigna el color verde al boton
set(handles.pushbutton14,'backgroundcolor','g'); % Asigna el color verde al boton
if (data_att(j,1)>90)
set(handles.pushbutton16,'backgroundcolor','g'); % Asigna el color verde al boton
end

%Delay to display graph
data_att(j,2) =
calllib('thinkgear64','TG_GetValue',connectionId1,TG_DATA_MEDITATION );
%%%%%%%PAUSA
data_att(j,3) = calllib('thinkgear64','TG_GetValue',connectionId1,TG_DATA_RAW );
blink_senal=abs(data_att(j,3)); %valor absoluto
blink=length(blink_senal(blink_senal>600)); %valor de parpadeo mayor que 600
%Mostrar la venta de comandos
str=blink_senal;
str1=num2str(str); % convierte en valores números
set(handles.edit3,'String',str1);
disp(data_att(j,3));
%Graficar las señales en axe
axes(handles.axes3);
plot(data_att(:,3),'g','Linewidth',1);
title('Parpadeo');
%%%%
if (blink_senal<600)) % compara los datos menores que 600
  if(aux==1)
set(handles.pushbutton2,'backgroundcolor','g'); % Asigna el color verde al boton
set(handles.pushbutton5,'backgroundcolor','w'); % Asigna el color verde al botón
set(handles.pushbutton4,'backgroundcolor','w'); % Asigna el color verde al botón
set(handles.pushbutton6,'backgroundcolor','w'); % Asigna el color blanco al botón
set(handles.pushbutton3,'backgroundcolor','w'); % Asigna el color blanco al botón
set(handles.pushbutton25,'backgroundcolor','w'); % Asigna el color blanco al botón
str=data_att(j,1); %adquiere los datos de atención en la variable str
str1=num2str(str); % convierte en valores números
set(handles.edit8,'String',str1);
%Mostrar la venta de comandos
disp(data_att(j,1));
%Graficar las señales en axe
axes(handles.axes1);
plot(data_att(:,1),'r','Linewidth',1);
title('IZQUIERDO');
if(data_att(j,1)>=70) % compara los datos mayores e iguales que 70
```

```
%  str=data_att(j,1); %adquiere los datos de atención en la variable str
% str1=num2str(str); % convierte en valores números
com=com+1;
com1=0; Asignada el valor 0 a la variable com1
com2=0; Asignada el valor 0 a la variable com2
com3=0; Asignada el valor 0 a la variable com3
com5=0; Asignada el valor 0 a la variable com5
com4=0; Asignada el valor 0 a la variable com4
%  asignar valores
set(handles.edit14,'String',com);
set(handles.edit16,'String',' '); % Asigna un  carácter  vacío
set(handles.edit17,'String',' '); % Asigna un  carácter  vacío
set(handles.edit17,'String',' '); % Asigna un  carácter  vacío
set(handles.edit19,'String',' ');
set(handles.edit21,'String',' '); % Asigna un  carácter  vacío
if (com==2)
baseiz
end

%Mostrar la venta de comandos
end
%   aux=aux+1;
else
if(aux==2)
%%subir%%%%%%%%%%%%%%
% Colorear los bonotes
set(handles.pushbutton5,'backgroundcolor','g'); % Asigna el color blanco al botón
set(handles.pushbutton2,'backgroundcolor','w'); % Asigna el color blanco al botón
set(handles.pushbutton4,'backgroundcolor','w'); % Asigna el color verde al botón
set(handles.pushbutton6,'backgroundcolor','w'); % Asigna el color blanco al botón
set(handles.pushbutton3,'backgroundcolor','w'); % Asigna el color blanco al botón
set(handles.pushbutton25,'backgroundcolor','w'); % Asigna el color blanco al botón
str=data_att(j,1); %adquiere los datos de atención en la variable str
str1=num2str(str); % convierte en valores números
set(handles.edit10,'String',str1);
%Mostrar la venta de comandos
disp(data_att(j,1));
%Graficar las señales en axe
axes(handles.axes5);
plot(data_att(:,1),'b','Linewidth',1);
title('SUBIR');
if(data_att(j,1)>=70) % compara los datos mayores e iguales que 70
%  str=data_att(j,1); %adquiere los datos de atención en la variable str
% str1=num2str(str); % convierte en valores números
com1=com1+1;

com=0; Asignada el valor 0 a la variable com
com2=0; Asignada el valor 0 a la variable com2
com3=0; Asignada el valor 0 a la variable com3
com5=0; Asignada el valor 0 a la variable com5
com4=0; Asignada el valor 0 a la variable com4

set(handles.edit17,'String',com1);
%Mostrar la venta de comandos
```

```
if (com1==2)
sube
end
end

%    aux=aux+1;
else
if(aux==3)
%%%%derecho%%%%%
set(handles.pushbutton4,'backgroundcolor','g');
set(handles.pushbutton6,'backgroundcolor','w'); % Asigna el color blanco al botón
set(handles.pushbutton3,'backgroundcolor','w'); % Asigna el color blanco al botón
set(handles.pushbutton2,'backgroundcolor','w'); % Asigna el color blanco al botón
set(handles.pushbutton5,'backgroundcolor','w'); % Asigna el color verde al botón
set(handles.pushbutton25,'backgroundcolor','w'); % Asigna el color blanco al botón
str=data_att(j,1); %adquiere los datos de atención en la variable str
str1=num2str(str); % convierte en valores números
set(handles.edit9,'String',str1);
%Mostrar la venta de comandos
disp(data_att(j,1));
%Graficar las señales en axe
axes(handles.axes2);
plot(data_att(:,1),'r','Linewidth',1);
title('DERECHO');
if(data_att(j,1)>=70) % compara los datos mayores e iguales que 70
 com2=com2+1;
com1=0; Asignada el valor 0 a la variable com1
com=0; Asignada el valor 0 a la variable com
com3=0; Asignada el valor 0 a la variable com3
com5=0; Asignada el valor 0 a la variable com5
com4=0; Asignada el valor 0 a la variable com4
set(handles.edit16,'String',com2);
set(handles.edit14,'String','  '); % Asigna un  carácter  vacío
set(handles.edit17,'String','  '); % Asigna un  carácter  vacío
set(handles.edit18,'String','  '); % Asigna un  carácter  vacío
set(handles.edit19,'String','  ');
set(handles.edit21,'String','  '); % Asigna un  carácter  vacío
%Mostrar la venta de comandos
if (com2==2)
basedere
end
end
%aux=aux+1;
else
if(aux==4)
%%%%%%%%%%%% bajar%%%%%%%%%%%%
set(handles.pushbutton6,'backgroundcolor','g');
set(handles.pushbutton4,'backgroundcolor','w'); % Asigna el color verde al botón
set(handles.pushbutton3,'backgroundcolor','w'); % Asigna el color blanco al botón
set(handles.pushbutton2,'backgroundcolor','w'); % Asigna el color blanco al botón
set(handles.pushbutton5,'backgroundcolor','w'); % Asigna el color verde al botón
set(handles.pushbutton25,'backgroundcolor','w'); % Asigna el color blanco al botón
str=data_att(j,1); %adquiere los datos de atención en la variable str
str1=num2str(str); % convierte en valores números
```

```
set(handles.edit11,'String',str1);
%Mostrar la venta de comandos
disp(data_att(j,1));
%Graficar las señales en axe
axes(handles.axes4);
plot(data_att(:,1),'b','Linewidth',1);
title('BAJAR');
% aux=aux+1;
if(data_att(j,1)>=70) % compara los datos mayores e iguales que 70
%  str=data_att(j,1); %adquiere los datos de atención en la variable str
% str1=num2str(str); % convierte en valores números
com3=com3+1;
com1=0; Asignada el valor 0 a la variable com1
com2=0; Asignada el valor 0 a la variable com2
com5=0; Asignada el valor 0 a la variable com5
com=0; Asignada el valor 0 a la variable com
com4=0; Asignada el valor 0 a la variable com4

set(handles.edit18,'String',com3);
set(handles.edit14,'String','  '); % Asigna un  carácter  vacío
set(handles.edit16,'String','  '); % Asigna un  carácter  vacío
set(handles.edit17,'String','  '); % Asigna un  carácter  vacío
set(handles.edit19,'String','  ');
set(handles.edit21,'String','  '); % Asigna un  carácter  vacío
if (com3==2)
baja
end
%Mostrar la venta de comandos
end
else

if(aux==5)
%%%% pinza%%%%%%%%%%%%
set(handles.pushbutton3,'backgroundcolor','g');
set(handles.pushbutton6,'backgroundcolor','w'); % Asigna el color blanco al botón
set(handles.pushbutton4,'backgroundcolor','w'); % Asigna el color verde al botón
set(handles.pushbutton2,'backgroundcolor','w'); % Asigna el color blanco al botón
set(handles.pushbutton5,'backgroundcolor','w'); % Asigna el color verde al botón
set(handles.pushbutton25,'backgroundcolor','w'); % Asigna el color blanco al botón
str=data_att(j,2); % adquiriendo los valores de atención en la variable srt
str1=num2str(str); % convierte en valores numéricos
set(handles.edit12,'String',str1);
%Mostrar la venta de comandos
disp(data_att(j,2));
%Graficar las señales en axe
axes(handles.axes6);
plot(data_att(:,2),'r','Linewidth',1);
title('PINZA Abrir');
if(data_att(j,2)>=70) % compara los datos mayores e iguales que 70
%  str=data_att(j,1); %adquiere los datos de atención en la variable str
% str1=num2str(str); % convierte en valores numéricos
com4=com4+1;
com1=0; Asignada el valor 0 a la variable com1
com5=0; Asignada el valor 0 a la variable com5
```

```
com2=0; Asignada el valor 0 a la variable com2
com3=0; Asignada el valor 0 a la variable com3
com=0; Asignada el valor 0 a la variable com

set(handles.edit19,'String',com4);
set(handles.edit14,'String',' '); % Asigna un carácter vacío
set(handles.edit16,'String',' '); % Asigna un carácter vacío
set(handles.edit17,'String',' '); % Asigna un carácter vacío
set(handles.edit18,'String',' '); % Asigna un carácter vacío
set(handles.edit21,'String',' '); % Asigna un carácter vacío
if (com4==2)
pinzaa
end
end
else
if(aux==6)
%%%% pinza%%%%%%%%%%%%
set(handles.pushbutton25,'backgroundcolor','g');
set(handles.pushbutton3,'backgroundcolor','w'); % Asigna el color blanco al botón
set(handles.pushbutton6,'backgroundcolor','w'); % Asigna el color blanco al botón
set(handles.pushbutton4,'backgroundcolor','w'); % Asigna el color verde al botón
set(handles.pushbutton2,'backgroundcolor','w'); % Asigna el color blanco al botón
set(handles.pushbutton5,'backgroundcolor','w'); % Asigna el color verde al botón
str=data_att(j,2); % adquiriendo los valores de atencion en la variable srt
str1=num2str(str); % convierte en valores números
set(handles.edit20,'String',str1);
%Mostrar la venta de comandos
disp(data_att(j,2));
%Graficar las señales en axe
axes(handles.axes6);
plot(data_att(:,2),'r','Linewidth',1);
title('PINZA Cierre');
if(data_att(j,2)>=70) % compara los datos mayores e iguales que 70
%  str=data_att(j,1); %adquiere los datos de atención en la variable str
% str1=num2str(str); % convierte en valores números
com5=com5+1;
com1=0; Asignada el valor 0 a la variable com1
com2=0; Asignada el valor 0 a la variable com2
com3=0; Asignada el valor 0 a la variable com3
com4=0; Asignada el valor 0 a la variable com4
com=0; Asignada el valor 0 a la variable com

set(handles.edit21,'String',com5);
set(handles.edit14,'String',' '); % Asigna un carácter vacío
set(handles.edit16,'String',' '); % Asigna un carácter vacío
set(handles.edit17,'String',' '); % Asigna un carácter vacío
set(handles.edit18,'String',' '); % Asigna un carácter vacío
set(handles.edit19,'String',' ');
if (com5==2)
pinzac
end
end
```

```
%
else
aux=0;
end

end
end
end
end
end
else
aux=aux+1;
end
%%%%
%pause(1);
end
end
end

%Mostrar la venta de comandos
disp('Loop Completed')
%Desconecta los puertos de comunicaciones
calllib('thinkgear64', 'TG_FreeConnection', connectionId1 );
```

Descripción

En la primera parte del código se selecciona el puerto de comunicación del casco, la velocidad en baudios para usar la librería thinkgear dll , posteriormente se escoge el tipo de dato que solicita la librería, como siguiente paso carga la librería verifica e identifica la conexión, establece los archivos de registro de flujos para la conexión registra los datos e ingresa a un bucle, lee los datos proporcionados por las señal TG_DATA_ATTENTION, TG_DATA_MEDITATION, TG_DATA_RAW, TG_DATA_BLINK_STRENGTH. La señale original RAW proporciona datos positivos y negativos en un rango de 0 a 1000 µv, se aplicó a la señal el valor absoluto con la finalidad de operar con los valores positivos, posteriormente se realizó el algoritmo para la selección de movimientos donde se filtró los datos que va en el intervalo de (0-600) µV generando un pulso de selección de esta forma se obtiene el desplazamiento de los iconos de control como se puede apreciar en la figura 59, luego de seleccionar un movimiento se lee los datos de la señale TG_DATA_ATTENTION, se filtra los datos positivos en el intervalo (80-100) µV, para generar el pulso de activación se registra 20 datos, que superen valores mayores a 80 µV se esta forma podemos asegurar que la persona tiene le control de la señal de atención. La señal de meditación (TG_DATA_MEDITATION) es inversas a la señal de atención ya que la

señal meditación sube y la de atención baja, se utilizó en el control de la pinza (Gripper)

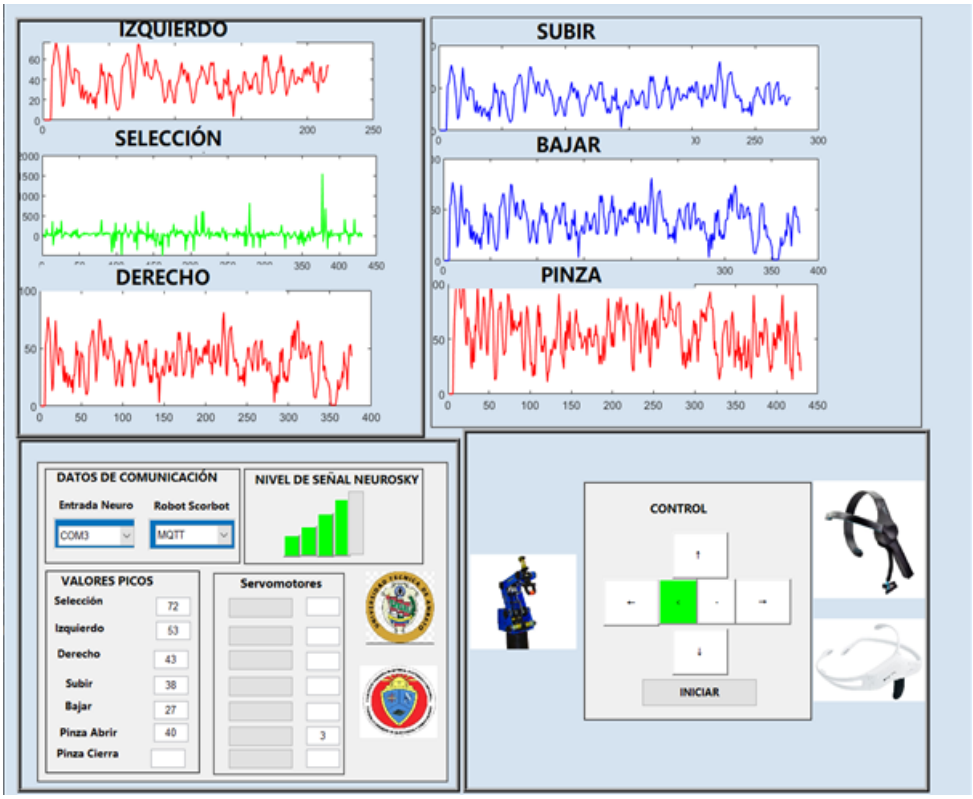


Figura 65: Interfaz del proyecto

Elaborado por: El investigador

Anexo 6: Método de comunicación mqtt en matlab

```
%Añade las librerias Mqtt eclip,oaho.cliente
javaaddpath('jars/iMqttClient.jar')
%Se estable el tópico casa para el método
PUB_TOPIC_1 = 'casa';

% Se añade una dirección ip para el método
mqttinterface = MqttInterface('matlab_mqtt_node', '172.88.1.104', 1883, 10);
% Mensaje del tópico
 pub_msg_1 = '1';
 % Valores predeterminados con el valor (0) de qos
 mqttinterface.send(PUB_TOPIC_1, pub_msg_1);
  %    Confidurado como publicador
 disp(['Published ', pub_msg_1, ' to topic ', PUB_TOPIC_1])
```

Descripción

En el código se configuro el método para el envió de los datos como publicador con la dirección 172.88.1.104 con un tópico casa por el puerto 1883.

Anexo 7: Configuración del protocolo MQTT en Python para el control del Robot Scorbot

```
import RPi.GPIO as GPIO % importa las librerías
import os
import time % librería de tiempo
import json
in1 = 24 % Inicialización de variables
in2 = 23
in3 = 27
in4 = 22
in5 = 16
in6 = 20
ena = 25
enb = 18
enc = 19
temp1=1
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(in1,GPIO.OUT) % Inicializando las variables como salida
GPIO.setup(in2,GPIO.OUT)
GPIO.setup(in3,GPIO.OUT)
GPIO.setup(in4,GPIO.OUT)
GPIO.setup(in5,GPIO.OUT)
GPIO.setup(in6,GPIO.OUT)
GPIO.setup(ena,GPIO.OUT)
GPIO.setup(enb,GPIO.OUT)
GPIO.setup(enc,GPIO.OUT)
pwm_a=GPIO.PWM(ena,500) % Asignando duty de 500
pwm_b=GPIO.PWM(enb,500)
pwm_c=GPIO.PWM(enc,500)
pwm_a.start(40) % Inicializando pwm en 40 %
pwm_b.start(40)
pwm_c.start(40)

# Parametros para la conexión
topicolee = “casa”
servidormqtt = “172.88.1.104”

# BLOQUE BÁSICO DE PROGRAMA
def on_connect(client, userdata, flags, rc):
    print(“Conexión/código de resultado: “+str(rc))
    # Inicio o renovación de subscripción
    client.subscribe(topicolee)

# el tópico tiene una publicación
def on_message(client, userdata, msg):
```

```
    global global_var
    print(msg.topic+" "+str(msg.payload))
    global_var = msg.payload
    unmensaje = msg.topic+" "+str(msg.payload)
    # rutinas de movimiento
if global_var == b'1':
        print("run")
        GPIO.output(in1,GPIO.HIGH)
        GPIO.output(in2,GPIO.LOW)
        print("correr")
    if global_var == b'2':
        print("run")
        GPIO.output(in1,GPIO.LOW)
        GPIO.output(in2,GPIO.HIGH)
        print("correr")
    if global_var == b'0':
        print("STOP")
        GPIO.output(in1,GPIO.LOW)
        GPIO.output(in2,GPIO.LOW)
        GPIO.output(in3,GPIO.LOW)
        GPIO.output(in4,GPIO.LOW)
        print("correr")
    if global_var == b'3':
        print("run")
        GPIO.output(in3,GPIO.HIGH)
        GPIO.output(in4,GPIO.LOW)
        print("correr")
     client = mqtt.Client()
client.on_connect = on_connect  % estableciendo conexión
client.on_message = on_message % recepción de mensaje
client.connect(servidormqtt, 1883, 60) % estableciendo conexión con servidor mqtt
client.loop_forever()
```

Descripción

En primera instancia la declaración librerías, variables privadas y variables globales, luego se configura la instancia del cliente se añade la dirección 172.88.1.104 que escucha por el puerto 1883, se añade el tópico "casa", se inicializa el PWM en las variables ena, enb, enc, dependiendo de los caracteres que ingresen se ejecuta las rutitas produciendo los movimientos de los servos motores.

Anexo 8: Configuración del protocolo MQTT en la ESP32 el control de la silla de ruedas.

```
#include <WiFi.h>
# include <PubSubClient.h>
const  char * ssid = “TP-LINK_4B8A” ;
const char* password = “12345678”;
const char* mqtt_server = “172.88.1.104”;
WiFiClient espClient;
PubSubClient client(espClient);
long lastMsg = 0;
char msg[50];
int value = 0;
// Motor A
int ENA = 26;
int IN1 = 33;
int IN2 = 32;
// Motor B
int ENB = 25;
int IN3 = 35;
int IN4 = 34;
void setup() {
   Serial.begin(9600);
 pinMode (ENA, OUTPUT);
 pinMode (ENB, OUTPUT);
 pinMode (IN1, OUTPUT);
 pinMode (IN2, OUTPUT);
 pinMode (IN3, OUTPUT);
 pinMode (IN4, OUTPUT);
  setup_wifi();
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);
}
void setup_wifi() {
```

```
  delay(10);
  // Conectamos a una red wifi
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
 Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}
//  Recepción del mensaje mediante el tópico
void callback(char* topic, byte* payload, unsigned int length) {
  Serial.print("Message arrived [");
  Serial.print(topic);
  Serial.print("] ");
  int i=0;
  String valor;
  int dato;
//  display.clear();
  String msg;
  for (int i = 0; i < length; i++) {
    Serial.print((char)payload[i]);
    //display.clear();
    msg += (char)payload[i];
  }
// convierte el dato string en un valor entero o numérico
  valor = (char)payload[0];
  dato = valor.toInt();
```

```
      Serial.println(“valor”);
      Serial.println(dato);
// condición para el movimiento de los motores
      if (dato == 1)
           {
      //Direccion motor A
 digitalWrite (IN1, HIGH);
 digitalWrite (IN2, LOW);
 digitalWrite (ENA, HIGH); //Velocidad motor A
 //Direccion motor B
 digitalWrite (IN3, HIGH);
 digitalWrite (IN4, LOW);
 digitalWrite (ENB, HIGH); //Velocidad motor B
        }
if (dato==2){
  //Direccion motor A
 digitalWrite (IN1, LOW);
 digitalWrite (IN2, HIGH);
 digitalWrite (ENA, HIGH); //Velocidad motor A
 //Direccion motor B
 digitalWrite (IN3, LOW);
 digitalWrite (IN4, HIGH);
 digitalWrite (ENB, HIGH); //Velocidad motor B
          }
 }
void reconnect() {
  // Loop until we're reconnected
  while (!client.connected()) {
    Serial.print(“Attempting MQTT connection...”);
    // Intentando conectarse
    uint64_t chipid = ESP.getEfuseMac(); // MAC address of ESP32
    uint16_t chip = (uint16_t)(chipid>>32);
    char clientid[25];
```

```
    snprintf(clientid,25,”WIFI-Display-%04X”,chip);
    if (client.connect(clientid)) {
      Serial.println(“connected”);
      // Una vez conectado, publique un anuncio ...
      //client.publish(“Say”, “-t 'hello world'“);
      // vuelvo a suscribirme
      client.subscribe(“casa”);

    } else {
      Serial.print(“failed, rc=“);
      Serial.print(client.state());
      Serial.println(“ inténtalo de nuevo en 5 segundos “);
      // Espere 5 segundos antes de volver a intentarlo
      delay(5000);
    }
  }
}
void loop() {
  if (!client.connected()) {
    reconnect();
  }
  client.loop();
  }
```

Descripción

El código del controlador ESP32 al igual que los anteriores se declara librerías, variables se configura la instancia del cliente añadiendo la dirección IP 172.88.1.104, el puerto de escucha 1883, el tópico “casa”, recepta los datos proporcionados del publicador compara con los métodos y ejecuta los movimientos de la silla.

Anexo 9: Código copilado en el Arduino mega

```
#include <SPI.h>
#include <Ethernet.h>
#include <PubSubClient.h>
# include <Servo.h>
Servo base;
int ii;
int j=0;
int aux = 1;
  int aux2 = 1;
Servo base;
Servo dere;
Servo izqui;
Servo pinza;
base.write(0);
dere.write(0);
izqui.write(180);
pinza.write(150);
//#define E1 10  // Enable Pin for motor 1
//#define I1 8     // Control pin 1 for motor 1
//#define I2 9     // Control pin 2 for motor 1
// Direccion MAC
byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
// IP del servidor
IPAddress mqtt_server(172, 88, 1, 104);
// Topic con el que trabajamos
const char* topicName = “casa”; // Nombre del tópico
EthernetClient ethClient; // Crea un cliente que puede conectarse a una dirección IP de Internet
PubSubClient client(ethClient); // Crea una instancia de cliente sin inicializar.
// Encabezado de la función de devolución de llamada
void callback(char* topic, byte* payload, unsigned int length) {
// imprime el código recibido
  Serial.print(“Message arrived [“);
  Serial.print(topic);
  Serial.print(“] “);
  int i=0;
  String valor;
  int dato;
    for (i=0;i<length;i++) {
    Serial.print((char)payload[i]);    // Primero convertí Payload a int / float para manejarlo más tarde en el código
  }
  valor = (char)payload[0];
```

```
 dato = valor.toInt();
 Serial.println("valor");
 Serial.println(dato);
 if (dato == 1)
{
   motor1();
  delay(5000);

 }
 if (dato == 2)
 {
      motor11();     // Arrancamos
      delay(5000);
               }
       }
void setup()
{
 Serial.begin(9600); //Inicialización del Puerto serial para conexión
   base.attach(9);
  base.write(0);

 if (Ethernet.begin(mac) == 0) { // Inicializa la conexión ethernet
  Serial.println("Failed to configure Ethernet using DHCP");
 }
 client.setServer(mqtt_server, 1883);
 client.setCallback(callback);
  // for (int i = 8 ; i<11 ; i++)                  // Se declaró los pines
   //   pinMode( i, OUTPUT);
 }
void loop()
{
 if (!client.connected()) { // Cliente conectándose
    Serial.print("Connecting ...");
         if (client.connect("rece_arduino")) {
      Serial.println("connected"); // Mostrando mensaje conectado
      client.subscribe(topicName); // suscribiéndose al tópico
    } else {
      delay(5000);
    }
 }

 // Cliente a la escucha
 client.loop();
}

case 1:
```

```
//variación de la posición servomotor base
for (bas = 0; bas <= 90; bas += 1)
{
base.write(bas);
delay(15); // retraso de 15
}
break;
case 2:
for (bas = 0; bas <= 180; bas += 1)
80
{
base.write(bas);
delay(15); // retraso de 15
}
break;
case 3:
for (bas = 90; bas <= 180; bas += 1)
{
base.write(bas);
delay(15); // retraso de 15
}
break;
case 4:
for (bas = 180; bas >= 0; bas -= 1)
{
base.write(bas);
delay(15); // retraso de 15
}
break;
case 5:
for (bas = 90; bas >= 0; bas -= 1)
{
base.write(bas);
delay(15); // retraso de 15
}
break;
case 6:
for (bas = 180; bas >= 90; bas -= 1)
{
base.write(bas);
delay(15); // retraso de 15
}
break;
81
//variación de la posición servomotor Brazo Base
case 7:
```

```
for (der = 0; der <= 60; der += 1)
{
dere.write(der);
delay(15); // retraso de 15
}
break;
case 8:
for (der = 60; der >= 0; der -= 1)
{
base.write(der);
delay(15); // retraso de 15
}
break;

//variación de la posición servomotor Antebrazo
case 9:
for (izq = 180; izq >= 120; izq -= 1)
{
izqui.write(izq);
delay(15); // retraso de 15
}
break;
case 10:
for (izq = 120; izq <= 180;izq += 1)
{
izqui.write(izq);
delay(15); // retraso de 15
}
break;
82
//variación la posición servomotor Pinza
case 11:
for (pin = 150; pin >= 120; pin -= 1)
{
pinza.write(pin);
delay(15); // retraso de 15
}
break;
case 12:
for (pin = 120; pin <= 150; pin += 1)
{
pinza.write(pin);
delay(15); // retraso de 15
  }
void  motor11(){
for (ii = 90 ; ii >= 0; ii -= 1 )
```

```
 {
   // Desplazamiento del al ángulo correspondiente
   base.write(ii);
   // Una pausa de 25ms
   delay(15); // retraso de 15
   }
}
```

Descripción

El código del Arduino para el control del brazo robótico Eezy, se declaró las librerías ethernet, servo, PubSubCliente posteriormente las variables la asignación de la mac del Arduino, la ip 172.88.1.104 y el tópico casa. Se adquiere los datos proporcionados del publicador en caracteres se procedió a convertir los datos de tipo Sting en datos numéricos con la finalidad de comparar con los métodos de ejecución de las rutinas del brazo.

Anexo 10: Configuración del software Unity 3D con el protocolo MQTT

```
 //Librerias
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using System.Net;
using System;
using uPLibrary.Networking.M2Mqtt;
using uPLibrary.Networking.M2Mqtt.Messages;
using uPLibrary.Networking.M2Mqtt.Utility;
using uPLibrary.Networking.M2Mqtt.Exceptions;
public class ROBOT : MonoBehaviour
{
// Declaración de las variables privadas para el cliente MQTT
private MqttClient client;
private string dat;
void Start () {
// Creamos la instancia del cliente
client = new MqttClient(IPAddress.Parse("172.88.1.104"),1883,false);
// se registra el mensaje que se recibe
client.MqttMsgPublishReceived += client_MqttMsgPublishReceived;
string clientId = Guid.NewGuid().ToString();
client.Connect(clientId);
// se subscribe al topico hello y se obtiene los datos
client.Subscribe(new string[] { "casa" }, new byte[] {
MqttMsgBase.QOS_LEVEL_EXACTLY_ONCE });
}
void client_MqttMsgPublishReceived(object sender, MqttMsgPublishEventArgs e)
{
Debug.Log("Received: " + System.Text.Encoding.UTF8.GetString(e.Message)  );
dat= System.Text.Encoding.UTF8.GetString(e.Message);
dato = Convert.ToInt32(dat);//se covierte en dato a variable int
}
// se declaró variables públicas para el movimiento del brazo Robotico
public float dato;
public float x;
public float y;
public float z;
public float timeCount = 0.0f;
public float timeCount1 = 0.0f;
public float timeCount2 = 0.0f;
public float timeCount3 = 0.0f;
public float timeCount4 = 0.0f;
public float timeCount5 = 0.0f;
```

```
public Transform RobotBase;
public Transform Brazo;
public Transform Antebrazo;
public Transform pdere;
public Transform pizq;
void Update(){
// según dependa el dato realizara los debidos movimientos en grados de cada parte del robot como corresponda
 switch (dato)
{
case 1:
// Estable una nueva rotación
RobotBase.rotation = Quaternion.Slerp (Quaternion.Euler(0,0,0),Quaternion.Euler(0,-90,0), timeCount);
// Incrementamos nuestro temporizador
timeCount = timeCount + Time.deltaTime*1;
timeCount1=0; // Se inicializa el contador en 0
timeCount2=0; // Se inicializa el contador en 0
timeCount3=0; // Se inicializa el contador en 0
timeCount4=0; // Se inicializa el contador en 0
timeCount5=0; // Se inicializa el contador en 0

y=-90;
break;

case 2:
// Estable una nueva rotación
RobotBase.rotation = Quaternion.Slerp (Quaternion.Euler(0,-90,0),Quaternion.Euler(0,0,0), timeCount1);
timeCount1 = timeCount1 + Time.deltaTime;
timeCount=0;
timeCount2=0;
timeCount3=0;
timeCount4=0;
timeCount5=0;
  y=0;
break;

case 3:
// Estable una nueva rotación
Antebrazo.rotation = Quaternion.Slerp (Quaternion.Euler(20,y,0),Quaternion.Euler(40,y,0), timeCount2);
timeCount2 = timeCount2 + Time.deltaTime*2;
timeCount=0;
timeCount1=0;
timeCount3=0;
timeCount4=0;
```

```
timeCount5=0;
break;
case 4:
// Estable una nueva rotación
Antebrazo.rotation = Quaternion.Slerp (Quaternion.Euler(40,y,0),Quaternion.Euler(20,y,0), timeCount4);
timeCount4 = timeCount4 + Time.deltaTime*2;
timeCount=0;
timeCount1=0;
timeCount2=0;
timeCount3=0;
timeCount5=0;
break;

case 5:
// Estable una nueva rotación
pdere.rotation = Quaternion.Slerp (Quaternion.Euler(-20,y+125,80),Quaternion.Euler(-20,y+110,80), timeCount3);
pizq.rotation = Quaternion.Slerp (Quaternion.Euler(-20,y+235,265),Quaternion.Euler(-20,y+250,265), timeCount3);
timeCount=0;
timeCount1=0;
timeCount2=0;
timeCount4=0;
timeCount5=0;
timeCount3 = timeCount3 + Time.deltaTime;

break;
case 6:
pdere.rotation = Quaternion.Slerp (Quaternion.Euler(-20,y+110,80),Quaternion.Euler(-20,y+125,80), timeCount5);
pizq.rotation = Quaternion.Slerp (Quaternion.Euler(-20,y+250,265),Quaternion.Euler(-20,y+235,265), timeCount5);
timeCount=0;
timeCount1=0;
timeCount2=0;
timeCount3=0;
timeCount4=0;
timeCount5 = timeCount5 + Time.deltaTime;

break;
default:
Debug.Log("no se obtiene datos");
break;
}
```

Descripción

El código representa a los movimientos del brazo robótico diseñado, en primera instancia la declaración librerías, variables privadas y variables globales, luego se configura el cliente se añade la dirección 172.88.1.104 que escucha por el puerto 1883, se añade el tópico “casa”, también dependiendo del dato que recibe ingresa a un condicional ejecutando las rutinas del brazo.